ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO TÉCNICO EN INFORMÁTICA DE GESTIÓN

Título del proyecto:

"SISTEMA DE INFORMACIÓN DE REPRESALIADOS DE LA GUERRA CIVIL EN NAVARRA"

Iosu Abaurrea Roldán

José Javier Astrain Escola

Pamplona, 17 de Julio de 2013

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INDICE

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INTRODUCCION

Punto de partida

El origen de este proyecto es el proyecto de investigación gallego "Nomes e Voces", creado en 2006 a través de un convenio entre 3 universidades gallegas y la consejería de Cultura de la Xunta de Galicia. Este proyecto tenía como objetivos el estudio de la represión durante la guerra civil y el franquismo, la recopilación de datos en una base de datos, y el mostrar información relativa a las víctimas.

Partiendo de esta idea, el grupo de investigación Fondo Documental de la Memoria Histórica en Navarra formado por Emilio Majuelo, Juan Carlos García, Gotxon Garmendia y Fernando Mendiola quiso trasladarla al ámbito de Navarra. Para ello pidieron ayuda al Servicio Informático de la Universidad Pública de Navarra, que se encargó brevemente de ello pero que tuvo que dejarlo debido a la falta de recursos para hacerle frente. A partir de este punto es cuando el proyecto se empieza a ofrecer como Proyecto Fin de Carrera para posteriormente hacerme cargo de él como mi PFC.

Una vez iniciado el proyecto se me facilitó todo el código fuente del proyecto gallego para tener una base sobre la que trabajar. Este código estaba hecho en PHPCAN, un framework de código abierto basado en php5. Así pues, el proyecto se realizaría sobre este lenguaje, añadiendo además HTML, PHP y JavaScript. Para la base de datos se empleará MySQL.

Para desarrollar el proyecto se siguieron dos líneas de trabajo. La primera consistía en desarrollar la interfaz de la web, adaptándola las nuevas necesidades funcionales del grupo de investigación, además de dotar a la web de una estética propia diferenciada del proyecto original. Para realizar esta tarea era imprescindible familiarizarse con phpCan, por lo que los cambios en la web comenzarían cuando el dominio de este lenguaje lo permitiese.

La otra línea de trabajo era el diseño de la base de datos en función de los requisitos planteados por los investigadores. Para ello, se realizaron reuniones periódicas con el grupo de investigación, con el fin de que la base de datos cubriese todas sus necesidades.

El objetivo final del proyecto es que el grupo de investigación dispusiera de un portal web totalmente ajustado a sus necesidades. Este portal web se dividirá en dos módulos, uno de acceso público y otro privado. El módulo de acceso público tendrá que permitir a cualquiera que acceda a él realizar búsquedas acerca de personas represaliadas en Navarra. Además deberá tener un apartado de noticias en los que el grupo de investigación de a conocer actividades y eventos en los que participa

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

El módulo privado será de uso exclusivo para los investigadores, y deberá permitirles manejar la base de datos en función de sus necesidades. Además los miembros del grupo de investigación podrán emplear consultas avanzadas sobre la base de datos para poder realizar estudios estadísticos sobre la información almacenada.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ANÁLISIS DE REQUISITOS

Objetivos generales

El proyecto sigue los mismos principios de su homólogo gallego, un portal web dividido en una parte pública y otra privada a la que solo se pueda acceder mediante el uso de nombre y contraseña. Ambas comparten una base de datos en la cual se recoge la información de la investigación.

En la parte pública se pueden consultar datos sobre personas represaliadas y ver las noticias que escriba el grupo de investigación. La parte privada está reservada para que los investigadores manejen la base de datos, añadan archivos de audio o fotos o creen noticias para la parte pública.

Por petición del grupo, el portal web estará en español, euskera e inglés pudiendo alternar entre ellos en cualquier momento.

Requisitos Funcionales

- El portal web deberá tener dos módulos, uno de libre acceso y otro privado.
- El módulo público deberá disponer de una página de presentación , una página con noticias relacionadas con el proyecto y un buscador de personas represaliadas
- La búsqueda se podrá hacer poniendo solamente el nombre de la persona o especificando los siguientes campos: Nombre, primer apellido, segundo apellido, municipio o localidad de nacimiento, municipio o localidad de residencia, lugar de cautiverio y afiliación sociopolítica.
- Al realizar una búsqueda se mostrará un listado de todas las personas que coincidan con los parámetros establecidos. Por defecto, si se accede a la página del buscador se mostrará una lista ordenada de todas las personas. Todas las personas que aparezcan en el listado incluirán una breve descripción.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- Al seleccionar una persona se mostrará una ficha con sus datos personales. Estos datos mostrarán:

    - Género.
    - Fecha y lugar de nacimiento (localidad, municipio y provincia).
    - Lugar de residencia.
    - Profesión.
    - Afiliación y cargo que ocupaba.
    - Nombre del padre, la madre y el cónyuge, así como el número hijos.
    - Fecha y lugar de su muerte.
    - Una breve descripción de su trayectoria.
    - Sucesos padecidos por esa persona.
    - Cautiverios padecidos.
    - Fotografías de esa persona.
    - Entrevistas que hablen de esa persona.

- El modulo privado permitirá la visualización y el manejo de todas la tablas de la base de datos, para ello podremos:
    - Añadir nuevas registros a la tabla.
    - Ver un listado de todos los registros que contiene la tabla, pudiendo elegir los campos de esta que queremos ver.
    - Editar cualquiera de los registros de la tabla.
    - Relacionar un registro de la tabla con registros de otras tablas en función de cómo esté establecida su relación.
    - Realizar búsquedas avanzadas mediante consultas SQL.

- Además el módulo privado dispondrá de una opción mediante la cual se podrán realizar las siguientes tareas de administración:
    - Actualizar la base de datos.
    - Revisión del sistema
    - Manejar los permisos de los usuarios (solo admin y superadmin).
    - Ver la actividad de los usuarios.
    - Cargar y descargar los archivos de idioma.
    - Descargar la base de datos.
    - Subir archivos mp3.
    - Trocear archivos mp3.

Requisitos no funcionales

- El módulo público debe tener una estética propia diferente a la del proyecto original e independiente de la de la universidad.
- Ambos módulos tienen que ser sencillos e intuitivos (especialmente el público), de forma que personas sin conocimiento informático puedan manejarlos.
- Todo el portal web tiene que estar disponible en castellano, euskera e inglés.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

HERRAMIENTAS

PhpCan

Es un framework escrito en php5 (gracias a la utilización de la versión 5 de php podemos implementar con facilidad la programación orientada a objetos) que facilita la construcción de webs gracias a una estructura ordenada basada en el sistema Modelo-Vista-Controlador. Además nos proporciona un conjunto de funciones, clases y "formatos" que nos permiten construir nuestro portal web evitando en gran medida programar a nivel de PHP o HTML. Los "formatos" (formats en el original) son los elementos que permiten a phpCan comunicarse con la base de datos. No hay que confundir un campo de la base de datos con un formato. Un campo es un único elemento de información mientras que un formato puede agrupar a varios campos.

PHP 5

PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página web resultante. La versión 5 de PHP aporta todas las características necesarias para usar la programación orientada a objetos, hecho de importancia capital para nosotros ya que phpCan está basado en este tipo de programación. Aunque la presencia de php5 en nuestro proyecto se debe sobre todo a la construcción de phpCan, también utilizaremos PHP para dar forma a nuestro portal web.

HTML 5

HTML (HyperText Markup Languaje) hace referencia al lenguaje de marcado predominante para la elaboración de sitios web que se utiliza para describir y traducir la estructura y la información en forma de texto, así como para complementar el texto con objetos tales como imágenes. La forma de escritura de HTML es mediante etiquetas rodeadas por corchetes angulares (<, >). Este lenguaje (junto con algunas funcionalidades de phpCan) nos permite describir la apariencia de nuestra web.

La versión 5 de HTML nos proporciona nuevos elementos para dar forma a nuestra web (por ejemplo footer, section o nav) que nos evitan la reiteración del elemento div. También añade mejoras en los formularios y facilidades para validar el contenido sin JavaScript.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

JavaScript

JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. Lo utilizaremos para manejar parte del comportamiento de nuestra interfaz gráfica y la reproducción de archivos de audio.

Css 3

Las hojas de estilo en cascada o Cascading Style Sheets (css) hacen referencia a un lenguaje de hojas de estilos usado para describir la presentación semántica (el aspecto y el formato) de un documento escrito en lenguaje de marcas. Su aplicación más común es dar estilo a las páginas web escritas en lenguaje HTML y XHTML, pero también puede ser aplicado a cualquier tipo de documentos CML, incluyendo SVG y XUL.

La novedad más importante que aporta CSS 3, de cara a los desarrolladores de webs, consiste en la incorporación de nuevos mecanismos para mantener un mayor control sobre el estilo con el que se muestran los elementos de las páginas, sin tener que recurrir a trucos que a menudo complicaban el código de las web.

MySQL

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB desarrolla MySQL como software libre en un esquema de licenciamiento dual. Sin embargo las empresas que quieran incorporarlo para su uso privado deben comprar una licencia que les permita su uso.

En nuestro proyecto MySQL será el sistema encargado de sostener nuestra base de datos, sin embargo phpCan incorpora métodos propios para crear, relacionar, poblar y modificar las diferentes tablas de nuestra base, por lo que no utilizaremos muchas de sus funciones para esta tarea.

Awk y flex

AWK es un lenguaje de programación para procesar datos basados en textos, ya sean fichero o flujos de datos. Por su parte Flex es un programa para generar analizadores léxicos.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Hemos creado un par de scripts en estos lenguajes de programación como scripts auxiliares para en caso de ser necesarios adaptar los ficheros de Excel que nos envíe el grupo de investigación para que se cargar bien en nuestra base de datos.

Para utilizar todos estos lenguajes de programación utilizaremos diferentes programas.

Aptana

Apatana es un entorno de desarrollo integrado de software libre basado en eclipse y desarrollado por Aptana. Inc, que puede funcionar bajo Windoes, Mac y Linux y provee soporte para lenguajes como: PHP, Python, Ruby, CSS o HTML. Dado todos los lenguajes que soporta y que permitía hacer una conexión ssh con nuestro servidor para hacer modificaciones directamente en él, utilizamos Aptana para toda la programación de la web.

WinSCP

WinSCP es una aplicación de Sofware libre. WinScp es un cliente SFTP gráfico para Windows que emplea SSH y que nos permitía subir archivos al servidor de la web desde nuestro ordenador.

Putty

Putty es un cliente SSH que nos permitía realizar estas conexiones en Windows a través de una consola. Esto resulta importante para el manejo de la base de datos que solo se podía manejar a través de comandos de mysql.

Poedit

Poedit es un editor que nos permite crear archivos .mo útiles para la traducción de nuestra web a partir de archivos .po creados por nosotros mismos para realizar la traducción.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DISEÑO

Interfaz gráfica

Aunque este proyecto partía de uno gallego, el grupo de investigación tenía claro que querían una estética propia en la parte pública del mismo. Para ello se creó la página de muestra que serviría de esquema para la versión final. A los investigadores les gustó el esquema, por lo que se empezó con lo que sería el diseño final de la web.

Este diseño está basado en una cabecera fija que se mantiene accesible desde cualquiera de los apartados a los que podemos acceder.

Para la página de presentación se optó por poner un texto fijo de introducción con una breve explicación del proyecto. Como el objetivo primordial de esta plataforma es ser un buscador de personas represaliadas, se le añadió un cajetín de búsqueda a la derecha del texto, de forma que sería muy sencillo empezar a realizar una búsqueda simple.

Figura 1. Interfaz final para la página de presentación

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

En la página del buscador optó por tener dos componentes: un cajetín para realizar búsquedas y que también estaría fijo en esta parte, y una sección donde mostrar los resultados de las búsquedas o los datos pormenorizados de una persona.

El cajetín a su vez dispondría de tres elementos: un campo en el que escribir el nombre de la persona a buscar, una pestaña desplegable para hacer una búsqueda avanzada y una pestaña con la opción de listar todas las personas.

La pestaña desplegable mostrará 8 parámetros que pueden ser rellenados: 3 campos en los que escribir nombre, primer apellido y segundo apellido; 5 pestañas desplegables en las que se podrá seleccionar lugar de nacimiento, lugar de residencia, sucesos padecidos, lugar de cautiverio y la afiliación sociopolítica de la persona.

Figura 2. Interfaz final del buscador avanzado

En la sección referente a los resultados, se mostrará un listado de las personas que coincidan con los resultados. Entre el listado y el cajetín se indicará el número de resultados obtenidos. Encabezando la muestra de personas se colocará una pestaña en la que seleccionar el modo de ordenar el listado, así como varios botones que permitan seleccionar cualquier página de todos estos resultados. Cada persona que aparezca tiene que tener una breve descripción en la cual aparezcan datos como nombre, apellidos, apodo, sexo, lugar y fecha de nacimiento, lugar de residencia así como indicar si la persona tiene alguna foto o archivo de audio asociados. Al pasar el cursor por cada ficha ésta cambiará de color para indicar que la estamos señalando.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

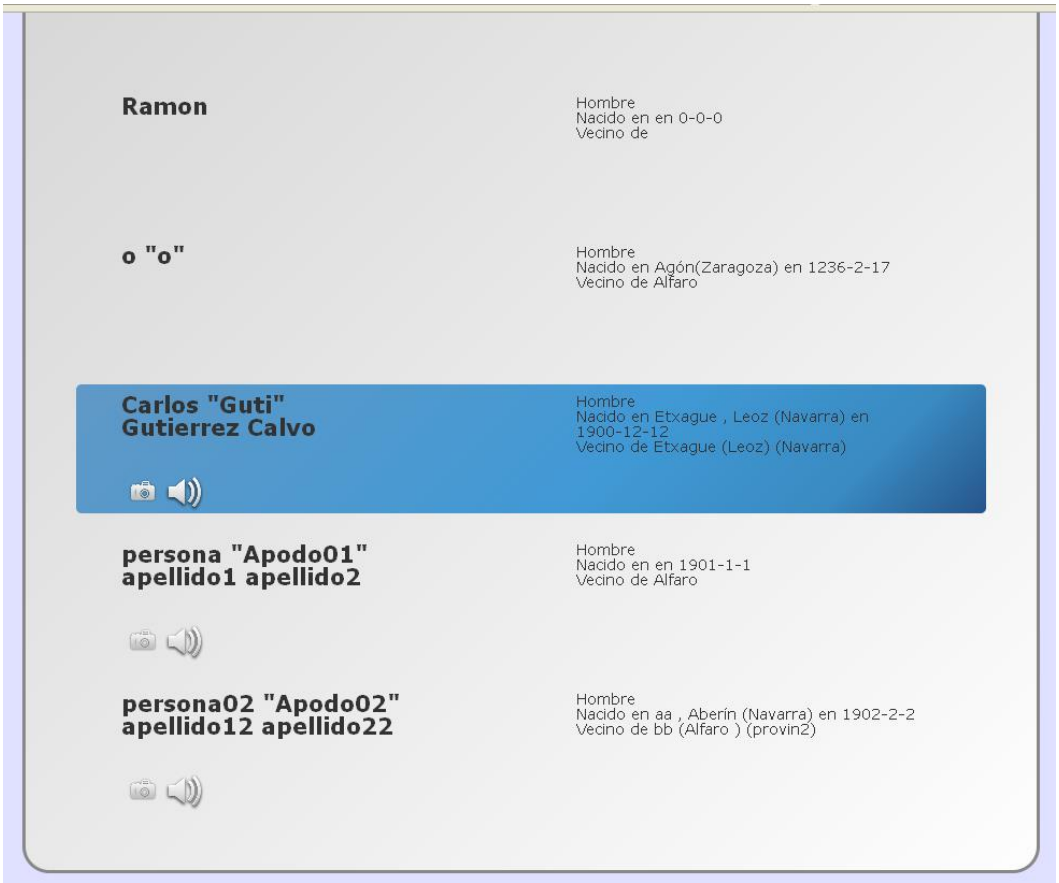


Figura 3. Interfaz final del listado de personas

La otra parte de la sección de resultados se da cuando seleccionamos una ficha concreta, entonces sustituimos la el listado por una ficha mucho más grande y detallada de la persona. En esta ficha tendremos los datos personales de la persona separados en dos columnas.

La columna de la izquierda comenzará con el nombre completo de la persona, resaltado en negrita y sensiblemente más grande que el resto de la ficha. A continuación de este se indicará el género de la persona, la fecha y lugar de nacimiento, el lugar de residencia, la profesión, su afiliación política, el cargo que ocupaba y una breve descripción de su trayectoria vital.

Por su parte, la columna de la derecha tendrá dos apartados, familia y datos de su muerte (en caso de que sea necesario). Estos dos apartados estarán separados en dos párrafos, cada uno de ellos encabezados por un título remarcado en negrita. En el párrafo de familia aparecerá una referencia a los padres de la persona y a su cónyuge,

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

además se indicará el número de hijos de esa persona. En el apartado de datos de muerte aparecerá el lugar y fecha de la muerte así como una breve explicación de los hechos.

Tras estas dos columnas aparecerá un listado de los sucesos padecidos por la persona. Éste dispondrá de su propio título tras el cual aparecerán los sucesos en los cuales se detallara el nombre de suceso, la fecha y el lugar, y además se mostrará una breve descripción del suceso. Cada suceso estará separado de los demás por un pequeño borde.

A continuación, y de forma similar a los sucesos aparecerán los registros de cautiverio sufridos por la víctima. Estos aparecen dentro de una tabla con tres campos, lugar de cautiverio, delito cometido y periodo de cautiverio.

La última zona de la ficha se vuelve a dividir en dos columnas, la columna de la izquierda contendrá las fuentes de las cuales se ha obtenido la información de la persona, y la de la derecha mostrará las fotos (si las hubiera) de la persona, así como los archivos de audio en los que se habla de ella.

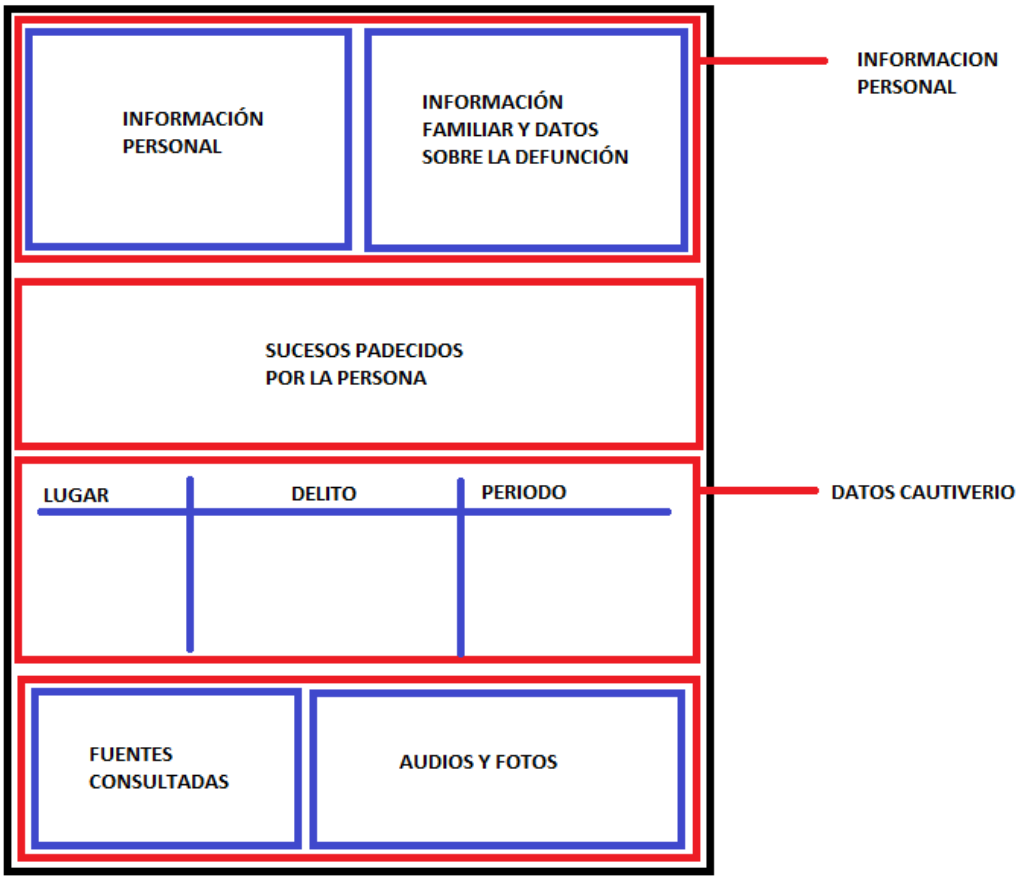


Figura 4. Estructura de la interfaz de la ficha detallada de usuario

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Base de datos

Para la base de datos se partió de la original del proyecto gallego. Esto se hizo así porque era una base que cubría prácticamente las necesidades de nuestro proyecto, siendo un buen punto de partida sobre el que comenzar. Además, los investigadores ya estaban familiarizados con esta base de datos, y su relación con los investigadores gallegos ayudaría a solventar posibles dudas.

La base de datos tiene varias peculiaridades. Una de ellas es que las tablas se dividen en dos grupos en diferentes. Esta división no es una división real de la base de datos, pero ayuda a distinguir las tablas en base a su funcionalidad. Por un lado tenemos las tablas de contenido, que nos permitirán recopilar la información de la investigación y cuyo contenido es variable en función de la información que añadan los investigadores.

Por otro lado tenemos las tablas de textos o auxiliares, que son las encargadas de almacenar contenidos complementarios que ayuden a rellenar las tablas. Estas tablas serán fijas una vez se tenga su versión final, y se corresponden a tablas en las que almacenaremos nombres de pueblos, profesiones, delitos o lugares.

La otra peculiaridad es que, en apariencia, se da mucha redundancia de datos en lo referente a datos personales como nombre, apellidos, fecha de muerte etc. Esto es debido a que en esta base de datos se guardan los contenidos de diversas fuentes, y estas pueden referirse a una misma persona. Esta redundancia no es negativa, si no que permite contrastar información entre las fuentes.

Teniendo en cuenta estas particularidades, se procedió a modificar la base de datos en función de las necesidades del grupo. Para ello se crearon nuevas tablas (País, Noticias y Localidades), se eliminaron (se comentó el código) tablas que ya no hacían falta (Persoaxes Vitimas, Prazas, Concejos y Comarcas Geográficas). Además se cambió el comportamiento de la tabla de Sucesos, pasando de ser una tabla de textos de carácter general (una lista de sucesos definidos) a una tabla de contenidos con cada registro relacionado con una persona. A parte de todo esto se realizaron multitud de cambios menores como añadir campos (el campo edad en varias tablas de contenido, testigos y signatura en la tabla de Registros de Defunción), renombrar tablas adecuándolos al uso que se les daría (Rexistro prisions pasó a llamarse Lugares de Encierro y posteriormente Registro Cautiverio) o traduciendo su nombre gallego al castellano (Rexistros Defuncions se cambió a Registros de Defunciones o Concellos se modificó a Concejos por poner unos ejemplos). En este punto se planteó la traducción de todas las tablas y sus campos al castellano, pero los investigadores desecharon la idea al no considerarla necesaria, ya que estaban familiarizados con la nomenclatura gallega. Tampoco consideraron conveniente la eliminación de campos inútiles al considerar que no impedían el manejo de la base de datos.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Privilegios

Tenemos cuatro niveles de privilegios, *usuario externo*, *usuario registrado*, *administrador* y *super administrador*.

Los usuarios externos son aquellos que no tienen ninguna implicación directa con el proyecto y por lo tanto no podrán acceder a la parte privada del portal web, limitando su acceso a la parte pública.

Los usuarios registrados son los miembros pertenecientes al grupo de administrador. Pueden acceder a la parte privada haciendo uso de su nombre y contraseña personales. Dentro del módulo privado pueden trabajar con las tablas añadiendo y modificando elementos, pero no pueden acceder a las funcionalidades avanzadas de la página.

El administrador se corresponde con el desarrollador del proyecto y tiene permisos para prácticamente cualquier funcionalidad del módulo privado, estando restringido solamente a la hora de modificar los permisos de otros usuarios.

El super administrador es la persona del Servicio Informático de la UPNA que da soporte al proyecto. No tiene ninguna limitación con los permisos, pudiendo otorgarlos y quitarlos en función de las necesidades.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

IMPLEMENTACIÓN

Instalación

Se ha instalado CentOS 6.2 para la arquitectura x86_64 con las siguientes características:

- Instalación en inglés, teclado en español.
- Configuración manual de la tarjeta de red eth0
- Se ha creado una partición /boot(ext4) de 500 MB. En el resto del disco se crea un volumen LVM con una participación swap de 5968 y una participación raíz ext4 que ocupa el resto del disco.
- Se ha hecho instalación mínima añadiendo "MySQL Database Server", "PHP Support + php5-mysql" y "Web Server".
- Una vez acabada la instalación, se han añadido los siguientes paquetes:
  - Openssh-clients
  - Wget
  - Policycoreutils-python
  - System-config-firewall

Requerimientos técnicos

A partir del código fuente suministrado, los requerimientos técnicos del software necesario son:

- Servidor Web Apache 2.2.16 o superior con los siguientes módulos activos:
  - Env
  - Headers
  - Mime
  - Php5
  - Rewrite.

- Directivas:
  - AllowOverride All
  - Options-Indexes
  - AddDefaultCharset UTF-8

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- PHP versión 5.3.3 o superior con las siguientes extensiones activas:
  - Php5-cli
  - Php5-common
  - Php5-curl
  - Php5-dev
  - Php5-gd
  - Php5-imagick
  - Php5-mcrypt
  - Php5-mysql

- Directivas
  - Safe_mode=Off
  - Magic_quotes_gpc=Off
  - Magic_cuotes_runtime=Off
  - Max_execution_time=300
  - Max_input_time=600
  - Memory_limit= 500M
  - Error_reporting=E_ALL & ~E_NOTICE
  - Display_errors=Off
  - Log_errors=On
  - Register_globals=Off
  - Post_max_size=500M
  - Upload_max_filesize=500M
  - Allow_url_fopen=ON
  - Allow_url_iclude=Off
  - Default_socket_timeout=60

- MySQL version 5.1 o superior con la codificación de caracteres "utf8_unicode_ci".

Estructura inicial

Una vez hemos instalado phpCan en nuestro servidor nos encontramos con las siguientes carpetas y archivos:

- **.httacces** es un fichero de configuración para el servidor apache.
- **Index.php** es el fichero que empiezan cargando todas las páginas de nuestro sitio web.
- **File.php** es el fichero el cual cargan todos los ficheros dinámicos.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- **Phpcan** es la carpeta donde se almacenan todas las funciones del framework. Dentro de esta podemos encontrar los siguientes elementos:
    - **Router.php** es un fichero que determina la escena actual y las rutas base, esto es, determina en que parte de nuestro portal web nos encontramos. Por ejemplo determina si estamos en la parte de administración o no.
    - **Cache** es la carpeta donde se guardan todos los archivos almacenados en caché.
    - **Config** es la carpeta donde se encuentra la configuración básica de phpCan como las rutas a todos los directorios o las escenas activas.
    - La carpeta **includes** es donde se guardan todas las clases y funciones utilizadas por phpCan.
    - En la carpeta **logs** se guardan todos los registros (errores, registros php, registros de apache etc).
    - **Languages** es la carpeta donde se guardan los fichero que permiten realizar traducciones.

- La carpeta **web** es donde por defecto se almacenan los archivos de nuestro sitio web. Contiene las siguientes carpetas:
    - La carpeta **actions** en la cual se guardan los ficheros con acciones.
    - En la carpeta **config** almacenamos los archivos para configurar nuestro portal web, como los referentes a la conexión con la base de datos, rutas etc.
    - En la carpeta **data** se guardan todos los archivos que conllevan consultas a la base de datos.
    - En **includes** guardamos ficheros con funciones hechas por nosotros mismos (diferentes a las que se almacenan en **phpcan/includes** que son las que vienen hechas con phpcan).
    - **Languages** tiene la misma función que su carpeta homóloga de **phpcan.**
    - La carpeta **templates** es una de las más importantes ya que contiene los ficheros que dan forma a nuestra web.
    - En **uploads** tenemos los archivos los archivos que subimos a la página como fotos o audios.

- La carpeta **common** contiene los ficheros que se vayan a compartir entre varias escenas diferentes.

- Por último en la carpeta **modules** guardaremos los módulos que hayamos instalado.

Esta sería la estructura inicial nada más instalar phpCan, pero como hemos señalado anteriormente, nuestro proyecto se basa en otro proyecto anteriormente, por lo que se pueden apreciar diferencias entre nuestros archivos y los originales.

En primer lugar no utilizamos las carpetas **modules** ni **common.** Además disponemos de dos carpetas que hacen la función de la carpeta **web:** La carpeta buscador y la carpeta admin. La carpeta buscador tiene la misma estructura y se encuentra en el mismo lugar que la carpeta **web** original, podría decirse que

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

simplemente la hemos renombrado. En esta carpeta se encuentra todo lo relacionado con el módulo público de nuestro portal web. La carpeta admin se encuentra dentro de la carpeta **phpcan** y contiene todo lo relacionado con la parte privada. La única diferencia estructural entre estas dos carpetas es que la carpeta buscador contiene en la subcarpeta **config** los archivos para la configuración de todo el portal web tales como la el archivo de rutas, la conexión con la base de datos, la creación de las tablas o las característica básicas de la web.

Una vez aclarada la estructura de nuestros archivos vamos a explicar el proceso de implementación de nuestro sitio web.

Nociones básicas sobre phpCan

Aunque se base en php, phpCan difiere bastante su estructura como para que se comprenda su funcionamiento a simple vista. Para facilitar su compresión vamos a explicar el comportamiento y la utilidad de varias de las clases más presentes en la elaboración del portal web (obviamente seguiremos profundizando en phpcan a medida que avancemos en el desarrollo de la memoria). Como es un lenguaje orientado a objetos tendremos que crear instancias de nuestras clases para poder utilizarlas.

- **Clase Config:** Seguramente la clase más importante ya que se encarga de la configuración de todo tipo de archivos. En esta clase se almacenan en diferentes arrays datos acerca de multitud de otras clases como de las clases templates, db, sesión o relations. Debido a esto cuando las demás los métodos de las demás clases consultan la información en esta clase. Por ejemplo el método load de la clase templates ($templates->load()) buscará en la variable templates de la clase config el archivo que tiene que cargar.
- **Clase templates:** Aunque es una clase sencilla es muy importante para nosotros ya que como hemos visto en el ejemplo anterior se encarga de cargar los templates de nuestra web (los archivos que contienen el html que da forma a nuestro portal web).
- **Clase vars:** Esta clase es la encargada de guardar variables que indican si somos administradores o en que escena no encontramos por ejemplo. Permite a otras clases saber que archivos cargar en función del contenido de sus variables.
- **Clase html:** Esta clase es la encargada de generar código html. Por ejemplo mediante el método fomSelect se pueden crear una pestaña con contenido seleccionable y con el método a() se crean enlaces html.
- **Clase data:** Esta clase es la encargada de realizar las comunicaciones con la base de datos y guardar los resultados obtenidos.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Configuración

Lo primero es preparar los archivos que se van a encargar de la configuración del sitio web. Para ello disponemos de la clase "config" que permite cargar los diferentes archivos de la carpeta config mediante el método "load()". Este método distingue el dominio donde nos encontramos así que utilizara la ruta conveniente. Por ejemplo, al hacer $config->load('templates'); cargaremos los templates que se encuentran en buscador/config o en phpcan/admin/config en función de si nos encontramos en la parte de administración o en la parte pública de nuestra web.

En nuestra página nos encontramos con 3 carpetas de configuración, una general localizada en phpcan/ y otras 2 específicas para cada parte de la web. Para la parte púbica usaremos buscador/config (que también tiene algún archivo de uso general, como ya hemos comentado antes), y para la parte privada usaremos la carpeta phpcan/admin/config.

En la carpeta de configuración general tenemos 3 archivos: index.php, scenes.php, y paths.php. En el archivo index.php (figura 1) procedemos a cargar mediante el método "load()" los archivos de configuración para la comunicación con la base de datos (buscador/config/db.php), para la configuración de algunas funcionalidades básicas de la web (buscador/config/base.php), para la creación de las tablas de nuestra base de datos (buscador/config/tables.php, de este último hablaremos más adelante) y establecer los idiomas de nuestra web (buscador/config/languages.php) . Además en el archivo index se añade la extensión v1 a las rutas que hacen referencia a los templates.

```
$config->load('base', 'scene');
$config->load('db', 'scene');
$config->load('tables', 'scene');
$config->load('languages', 'scene');

//Templates path
// buscador/templates + /v1
$config->paths['www-scene-templates'] .= $config->base['template'].'/';
// phpcan/admin/templates/ + /v1
$config->paths['www-admin-templates'] .= $config->base['admin_template'].'/';
// data/www/memoriahistorica/buscador/templates + /v1
$config->paths['scene-templates'] .= $config->base['template'].'/';
// data/www/memoriahistorica/phpcan/admin/templates + /v1
$config->paths['admin-templates'] .= $config->base['admin_template'].'/';

//Debug init
$debug->start();

//Set language
$config->setLanguage();
```

Figura 5

El archivo db.php (figura 2) configura la conexión con nuestra base de datos estableciendo parámetros como el nombre o la contraseña.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```php
defined('ANS') or die();

$config->db = array(
    'enabled' => true,
    'type' => 'mysql',
    'host' => 'localhost',
    'database' => 'memoriahistorica',
    'user' => 'memoriahistorica',
    'password' => 'Takochdo-12',
    'prefix' => 'phpcan_', // Only for relations tables
    'persistent' => true
);
?>
```

Figura 6

El archivo base.php (figura 3) configura parámetros tales como la codificación de contraseñas, la versión de los templates o los lenguajes disponibles.

```php
defined('ANS') or die();

$config->base = array(
    'key' => md5(filemtime(__FILE__)), // Encrypt and decrypt key
    'debug' => true, // Show in screen debug errors, info and data
    'gettext' => true, // Gettext functions
    'languages' => array('eu','en','es'), // Available languages
    'language_detect' => 'subfolder', //Detect method for language (subfolder/subdomain/get)
    'template' => 'v1', // Template version
    'admin_template' => 'v1', // Admin template version

    'gmap_key' => ''
);
?>
```

Figura 7

El archivo laguages.php (figura 4) establece los lenguajes disponibles el lenguaje por defecto que tenemos en nuestra web.

```php
defined('ANS') or die();

$config->languages = array(
    'multilanguage' => true, // Enable/Disable gettext functions
    'detect' => 'subfolder', // Method for language detection (subfolder/subdomain/get)
    'default' => 'es', // If default is defined, load this language as default, else try to load the browser la
    'enabled' => array('es', 'eu', 'en'), // Enabled languages (public view) (ALWAYS ARRAY)
    'availables' => array('en', 'es', 'eu'), // Available languages (ALWAYS ARRAY)
    'admin' => array('en', 'es', 'eu'), // Admin languages (admin UI) (ALWAYS ARRAY)
);
?>
```

Figura 8

En los archivos scenes.php y paths.php definimos las escenas y las rutas pespectivamente (figuras 5 y 6).

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
*/

defined('ANS') or die();

$config->scenes = array(
    'vitimas' => array(
        'path' => $config->paths['web'].'buscador/',
        'templates' => 'buscador/templates/',
        'uploads' => 'buscador/media/',
        'detect' => 'subdomain'
    ),
);
```

Figura 9

```
defined('ANS') or die();

$base = dirname(getenv('SCRIPT_FILENAME'));
$base = str_replace('\\', '/', $base);
$config->paths = array(
    'server' => getenv('SERVER_NAME'),
    'web' => $base.'/',
    'cache' => $base.'/phpcan/cache/',
    'config' => $base.'/phpcan/config/',
    'formats' => $base.'/phpcan/formats/',
    'events' => $base.'/phpcan/events/',
    'includes' => $base.'/phpcan/includes/',
    'logs' => $base.'/phpcan/logs/',
    'admin' => $base.'/phpcan/admin/',

    'www' => preg_replace('#/+#', '/', '/'.preg_replace('|^'.getenv('DOCUMENT_ROOT').'|', '',
);

unset($base);
```

Figura 10

Hemos visto la carpeta “config” principal, ahora veremos las correspondientes a la parte pública y privada.

Como ya hemos comprobado en la explicación anterior, la carpeta /buscador/config no solo tiene archivos propios de la configuración de su módulo, sino que también tiene archivos correspondientes a la configuración de toda la web, pero ahora nos centraremos en los archivos de configuración específicos y que tiene en común con la carpeta /phpcan/admin/config. Tenemos 4 archivos comunes:

- El archivo templates.php (figura 7) se encarga de manejar los archivos que darán forma a nuestra página en cada subdirectorio. Estos archivos implican archivos del tipo php, css y js. Primero se declaran los archivos que se van a cargar siempre (por defecto) y después se añaden los archivos para cada situación específica (en el caso de la parte pública serían index, buscar, noticias y ficha). Por ejemplo si estando en el subdirectorio buscar realizamos un include de “$templates->load(‘contenido’)” estaremos incluyendo el archivo web_buscador.php, pero si lo hacemos desde el directorio index lo estaremos haciendo del archivo web_introduccion.php. El archivo data.php (figura 8) funciona de forma similar, preparando la carga de archivos concretos para momentos puntuales.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

- El archivo actions.php (figura 9) tienen un funcionamiento parecido a templates y data pero añade más atributos a tener en cuenta. A parte de cargar un archivo con para ejecutar una acción, podemos determinar detalles como re direccionar a otra página o refrescar la que estamos.
- El archivo sessions.php (figura 10) nos sirve para configurar la conexión del administrador con la base de datos al hacer login, ajustando parámetros como la tabla de la base de datos que contiene los usuarios, los campos de la tabla que hay que mirar para sacar el nombre y la contraseña, o como está codificada esta.

```
$config->templates = array(
    'default' => array(
        'html' => array(
            'css' => array(

                'styles.css',
                'stylesbuscador.css',
                'styleficha.css',
                'stylesnoticia.css'
            ),
            'js' => array(
                'jquery-1.3.2.min.js',
                'javascript-comun.js'
            ),

            'base' => 'baseweb.php',
        )
    ),

    'index' => array(
        'html' => array(
            'contenido'=>'web_introduccion.php'

        )
    ),

    'buscar' => array(
        'html' => array(

            'contenido'=>'web_buscador.php',
            'contenido_extendido'=>'web_listado.php'
        )
```

Figura 11

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
*/

defined('ANS') or die();

$config->data = array(
    'default' => array(
        'all' => 'menu.php'
    ),
    'undefined' => array(),

    'install' => array(
        'all' => 'install.php'
    ),

    'list-db' => array(
        'all' => 'list_db.php'
    ),
    'advanced-search' => array(
        'all' => 'advanced_search.php'
    ),
    'sql-search' => array(
        'all' => 'sql_search.php'
    ),
    'export-csv' => array(
        'html' => 'export_csv.php'
    ),
    'edit-db' => array(
        'all' => 'edit_db.php'
    ),
    'new-db' => array(
        'all' => 'new_db.php'
    ),
    'related-db' => array(
        'all' => 'related_db.php'
    )
```

Figura 12

```
defined('ANS') or die();

$config->actions = array(
    'default' => array(
        'disabled' => false,
        'check_file' => 'validate_action.php'
    ),

    'install' => array(
        'type' => 'admin',
        'file' => 'install.php',
        'check_file' => '',
        'reload' => path('login')
    ),

    'save-db' => array(
        'type' => 'table',
        'file' => 'save_db.php',
    ),
    'duplicate-db' => array(
        'type' => 'table',
        'file' => 'save_db.php',
    ),
    'delete_db' => array(
        'type' => 'table',
        'file' => 'delete_db.php',
        'reload' => true,
        'data' => '',
        'templates' => array(
            'ajax' => array(
                'base' => 'simple-ajax.php'
            )
        )
```

Figura 13

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
$config->session = array(
    'admin' => array(
        'enabled' => true, // Session contr
        'name' => 'phpcan_session', // Sess
        'maintain_time' => 3600 * 48, // Se
        'encrypt' => 'md5',

        'table' => 'users', // Session tabl

        'user_field' => 'user', // Session
        'password_field' => 'password', //
        'enabled_field' => 'active', // Ses
    )
```

Figura 14

Creación de la base de datos

A diferencia de lo que suele ser habitual a la hora de trabajar con una base de datos, con phpCan no se construyen tablas directamente, si no que la creación de estas son el fruto de definir el modelo entidad/relación. Esto es tenemos que definir las entidades con sus correspondientes atributos para más adelante establecer cómo serán las relaciones entre dichas entidades. Para ello utilizaremos los ficheros tables.php, tables-voces.php, tables-salamanca.php y tables-imaces.php. Aunque tenemos 4 ficheros para la definición de tablas en vez de un único fichero como se debería hacer, en la práctica solo cargaremos el fichero tables.php, que a su vez se encarga de cargar el resto de ficheros.

Lo primero que hay que tener en cuenta es que todo lo que decidamos en el fichero tables.php hay que guardarlo en su correspondiente atributo de la clase config, así que englobaremos todas las definiciones en un único array para que podamos hacerlo ($config-> tables=array) mientras que la declaración de relaciones las guardaremos en ($config-> relations=array).

En la figura 11 vemos un ejemplo de la declaración de entidades. Las entidades son arrays multidimensionales donde la primera dimensión se corresponde al nombre de la entidad, la segunda al nombre de los campos de dicha entidad y la tercera la ocupan los parámetros de cada campo. Los campos de estas entidades pueden tener varios parámetros diferentes, pero siempre tendrán que tener definido el parámetro de format (cuando solo hay un parámetro por defecto entiende que estamos definiendo el parámetro de format). Como ya hemos explicado antes los formats son un elemento especial de phpCan que nos permite almacenar hasta varios campos de una tabla. Por ejemplo un campo de cualquier tabla que queramos hacer multiidioma (definiendo el parámetro languages=>array(‘es’,’eu’) se representará como un solo campo en nuestra definición de entidad, pero en la base de datos aparecerá como dos campos cada uno con el mismo nombre de base pero uno acabado en _es y otro en _eu (nombre_es y nombre_eu). Otra característica de phpCan es que no usa campos propios de cada entidad para establecer la clave primeria sino que añade a cada tabla un campo id que es autoincremental y hace la función de clave primaria.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
defined('ANS') or die();

$config->tables = array(
    'ambitos' => array(
        'nome' => 'varchar'
    ),

    'causas' => array(
        'codigo' => array(
            'format' => 'varchar',
            'index' => true,
            'length_max' => 10
        ),
        'numero' => array(
            'format' => 'varchar',
            'length_max' => 10
        ),
        'feitos' => 'text',
        'inicio_de_causa' => array(
            'format' => 'date',
            'default' => '17-07-1936',
            'englishformat' => false
        ),
        'xurisdicion' => array(
            'format' => 'enum',
            'values' => 'Marina,Tierra'
        ),
        'fonte' => array(
```

Figura 15

Una vez definidas todas las entidades de nuestra base de datos, tenemos que establecer como se relacionan entre ellas. Para ello podemos ver en la figura 12 como cada relación está formada por un array que tiene como campos el campo tables, que indica las tablas que forman la relación; el campo mode que indica el tipo de relación que es (1-1, 1-x, x-1 o x-x); el campo name para poner nombre a la relación (necesario si una entidad se relaciona varias veces con la misma y por último el campo relation_table que se utiliza en las relaciones x-x si queremos crear una nueva tabla a partir de la relación.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
array(
    'tables' => 'ambitos registro_cautiverio',
    'name' => 'vecino_ambito',
    'mode' => '1 x'
),
array(
    'tables' => 'causas comarcas_geograficas',
    'mode' => 'x 1'
),
array(
    'tables' => 'causas municipios',
    'name' => 'municipio_delito',
    'mode' => 'x 1'
),
array(
    'tables' => 'causas localidades',
    'name' => 'localidad_delito',
    'mode' => 'x 1'
),
array(
    'tables' => 'causas persoas',
    'mode' => 'x x',
    'relation_table' => 'causa_por_persoa'
),
array(
    'tables' => 'causas prazas',
    'mode' => 'x 1'
),
array(
```

Figura 16

Como se ha dicho antes tenemos cuatro tipos de relaciones que afectan de formas diferentes a nuestra base de datos:

- Relación 1-1: En esta relación una entidad A se relaciona con una única entidad de B (un coche con su matrícula por ejemplo) esto conlleva la entidad A añada la clave principal de B a la hora de formar las tablas.
- Relacion 1-x: En esta relación una entidad A se relaciona con varias entidades de B, pero la entidad B solo se relaciona con una única entidad A. Esta relación conlleva que la entidad B se añada la clave principal de A a su tabla. La relación x-1 funciona exactamente igual intercambiando las entidades A y B.
- Relación x-x: En esta relación se da el caso de que varias entidades A se relacionan con varias entidades B. en este tipo de relación lo habitual es formar una tabla intermedia que tenga las claves primarias de A y B, pero phpCan utiliza una tabla llamada phpcan relations que relaciona los id de cada las tablas relacionadas entre sí. A pesar esto podemos crear una entidad que haga las veces de tabla intermedia y después hacer que la relación de las dos tablas se dé en esta tabla mediante el parámetro relation_table=>'nombre de la tabla'.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Primeros pasos

Ya hemos explicado cómo realizar la configuración de nuestro portal web y cómo definir la base de datos que la sustenta, ahora toca poner todo en funcionamiento.

Al igual que cualquier web hecha en PHP, nuestra página empieza cargando el archivo index.php (figura 13). En este caso, la función de este archivo es de iniciador, ya que solo se utiliza para llamar a más archivos necesarios para el funcionamiento de nuestra web. El archivo index llama mediante el método "require" de PHP a los archivos phpcan/index.php, phpcan/config/index.php y phpcan/router.php. Además utilizando el método load de la clase config nuestro archivo inicial llama a los archivos de configuración actions.php, events.php, data.php y templates.php. Estos archivos los cargará desde phpcan/admin/config o buscador/config en función de si estamos cargando la parte de administración o la parte pública. Además si nos encontramos en el módulo privado cargaremos también el archivo admin_tables.

```php
<?php
/**
* phpCan v.1
*
* $Id: index.php 2 2009-10-19 18:26:14Z lito $
*
* 2008 Copyright - phpCan is released under the GNU Affero GPL version 3
*
* More information at license.txt
*/
define('ANS', true);

error_reporting(E_ALL & ~E_NOTICE);

require ('phpcan/router.php');
require ($config->paths['config'].'index.php');

$config->load('actions');
$config->load('events');
$config->load('data');
$config->load('templates');

if ($vars->isAdmin()) {
    $config->load('admin_tables', 'scene');
}

require ($config->paths['includes'].'index.php');

$config->status = 'finished';
?>
```

Figura 17

El archivo router (figura 14) se encarga de crear una instancia de las clases vars, config y debug y de cargar los archivos phpcan/include/functions.php, phpcan/config/paths, phpcan/config/scenes.php y buscador/config/paths. El archivo functions.php se encarga de definir las funciones propias de phpcan, mientras que los archivos paths y scenes se encargan de definir las rutas de nuestra web.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
* 2008 Copyright - phpCan is released under the GNU Affero GPL version 3
*
* More information at license.txt
*/

defined('ANS') or die();

define('INCLUDES_PATH', dirname(__FILE__).'/includes/');
define('PHPCAN_VERSION', '1.0');

require_once (INCLUDES_PATH.'functions.php');

$vars = new Vars;
$config = new Config;
$debug = new Debug;

//carga los archivos paths.php y scenes.php

require_once (dirname(__FILE__).'/config/paths.php');
require_once ($config->paths['config'].'scenes.php');

$vars->load();
$config->setExitMode();
//completa el paths.php con el paths.php de buscador
require_once ($config->scenes[$vars->getScene()]['path'].'config/paths.php');

$vars->postLoad();

?>
```

Figura 18

El archivo config/index.php (figura 15) se ocupa de cargar los archivos de configuración necesarios para el funcionamiento de la web. Estos se encuentran en la carpeta buscador/config/ y son los archivos base.php, db.php, tables.php y languages.php. Como se ha explicado al principio del capítulo de configuración, la función de estos archivos es la de configurar características de la web y las bases de datos.

```
* More information at license.txt
*/

// phpcan/config/index.php

defined('ANS') or die();

//Common config (admin & scene)

// El path scene hace referencia a /data/www/memoriahistorica/buscador/
// esto esta definido en el archivo paths.php dentor de buscador/config
$config->load('base', 'scene');
$config->load('db', 'scene');
$config->load('tables', 'scene');
$config->load('languages', 'scene');

//Templates path
// buscador/templates + /v1
$config->paths['www-scene-templates'] .= $config->base['template'].'/';
// phpcan/admin/templates/ + /v1
$config->paths['www-admin-templates'] .= $config->base['admin_template'].'/';
// data/www/memoriahistorica/buscador/templates + /v1
$config->paths['scene-templates'] .= $config->base['template'].'/';
// data/www/memoriahistorica/phpcan/admin/templates + /v1
$config->paths['admin-templates'] .= $config->base['admin_template'].'/';

//Debug init
$debug->start();

//Set language
```

Figura 19

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

El archivo includes/index.php (figura 16) crea una instancia de las clases data, template y html, además carga los archivos que ayudan a la traducción de la web como son stream.php y gettext.php. Éstos se encuentran en phpcan/include/php-gettex. Los últimos archivos en cargar son basics.php (cargará el de buscador o el de phpcan/admin en función de donde nos encontremos), actions.php, data.php, templates.php.

```
* 2008 Copyright - phpCan is released under the GNU Affero GPL version 3
*
* More information at license.txt
*/
// phpcan/includes/index.php

defined('ANS') or die();

$data = new Data;
$templates = new Templates;
$html = new Html;

//Set controller
$vars->setController();

//Gettext
if ($config->languages['multilanguage']) {
    include_once ($config->paths['includes'].'php-gettext/streams.php');
    include_once ($config->paths['includes'].'php-gettext/gettext.php');
}

//Includes
if ($vars->isAdmin()) {
    if (is_file($config->paths['scene-includes'].'admin/basics.php')) {
        include_once ($config->paths['scene-includes'].'admin/basics.php');
    } else {
        include_once ($config->paths['admin-includes'].'basics.php');
    }
} else {
    // /data/www/memoriahistorica/buscador/includes/basics.php
```

Figura 20

El archivo basics se ocupa de cargar los archivos functions.php que se encuentran en phpcan/admin/includes o en buscador/includes con funciones hechas por nosotros mismo.

Los archivos actions.php, data.php y templates.php se encuentran en phpcan/include/ y su función es preparar los contenidos de sus respectivas variables de la clase config para que se pueda acceder de fácilmente a ellas. Además el archivo templates carga el archivo sobre el que se construye cada parte de nuestro portal web, este archivo se encuentra en phpcan/admin/templates/v1 en el caso de la parte de administración y en buscador/templates/v1 para la parte pública. Este es el archivo baseweb.php y es el archivo sobre el cual cargaremos el resto de archivos templates para dar forma a nuestra web.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Creación de la interfaz

Una vez hechas todas las labores de configuración vamos a comenzar la elaboración de la interfaz gráfica de nuestra web. Debido a limitaciones de tiempo en este proyecto solo se realizó la parte pública, por lo que la parte privada se reutilizó el código gallego aplicando cambios mínimos.

Como ya hemos mencionado antes, partimos de un único archivo sobre el que iremos añadiendo el resto de archivos con código HTML. Según se ha establecido en el diseño de la web, la parte pública constará de una cabecera y pie fijos, y un contenido principal variable en función del subdirectorio al que accedamos. Podemos ver en la figura 17 cómo será el esquema.

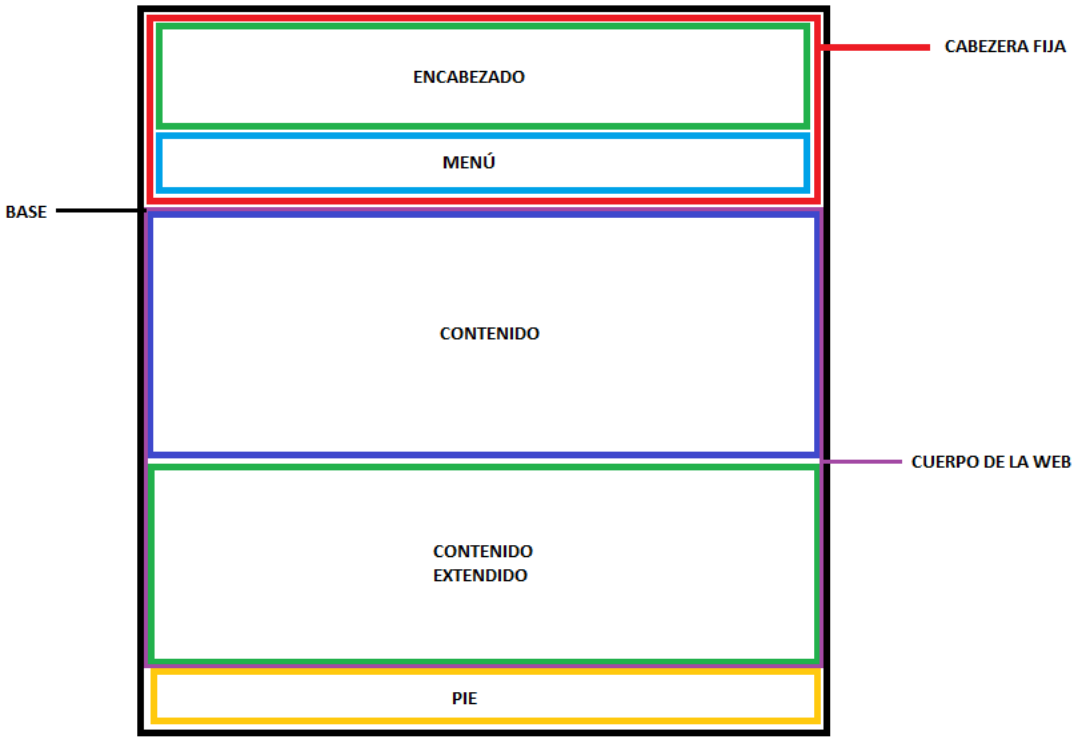


Figura 21

Al ser sólo el soporte donde se pondrá el resto de archivos, nuestro archivo baseweb.php es un archivo muy simple, con solo un par de elementos HTML (figura 18).

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    <!-- Head -->
    <?php include_once($templates->load('web_head.php')); ?>
    <!-- Fin Head -->

<body>
<div id="wrapper"><!-- #wrapper -->

    <?php include_once($templates->load('web_cabecera.php')); ?>

    <section id="main"><!-- #main content and sidebar area -->

                <?php include_once($templates->load('contenido')); ?>

                 <!-- <?php include_once($templates->load('contenido_extendido')); ?>

    </section><!-- end of #main content and sidebar-->



    <?php include_once($templates->load('web_pie.php')); ?>

</div><!-- #wrapper -->
<!-- Free template created by http://freehtml5templates.com -->
</body>
</html>
```

Figura 22

En la figura anterior podemos ver la utilización del método load. Este método busca el archivo solicitado en la variable $config->templates que hemos definido previamente en la carpeta buscador/config. Si encuentra el fichero devuelve la ruta completa hasta el archivo indicado por esta variable. Si por el contrario no encuentra el archivo en la variable, comprueba si el archivo existe en la carpeta buscador/templates/v1, si existe devuelve la ruta de este archivo, si no, devuelve la ruta de un archivo vacío (phpcan/includes/utils/empy_template.php). En la imagen anterior podemos ver como web_head.php y web_cabecera.php son archivos presentes en la carpeta templates y siempre cargarán lo mismo, mientras contenido y contenido extendido hacen referencia a los ficheros declarados en la variable $config->templates y por lo tanto devolverán un archivo u otro en función de la situación en la que nos encontremos. Poniendo un ejemplo real, si nos encontramos memoriahistorica.dyc.unavarra.es/index (memoriahistorica.dyc.unavarra.es es la dirección de la web) el método load('contenido') devolverá la ruta buscador/templates/v1/web_introduccion.php mientras que load('contenido_extendido') extendido no devolverá nada ya que el directorio index no tiene asignado ningún archivo para contenido_extendido.

Los archivos web_head.php y web_cabecera.php siempre se cargan ya que queremos mantener una cabecera fija. Al cargar web_head.php (figura 19) añadimos la cabecera de html, definimos el lenguaje por defecto y cargamos todos los archivos css y javascript que vamos a emplear (Esto último mediante el método $templates->headLinks()). A todo esto hay que sumar que definimos una pequeña función de javascript que nos permitirá extender y replegar la pestaña desplegable del buscador.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
    <!DOCTYPE html>
    <html lang="es">

<head>
    <meta charset="utf-8" />
    <title>Memoria Historica</title>
    <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
    <meta http-equiv="Content-Language" content="<?php echo $config->lang; ?>"
    <!-- <link rel="stylesheet" href="/buscador/templates/v1/css/styles.css" ty
    <?php echo $templates->headLinks(); ?>
    <script language="JavaScript">

        $(document).ready(function(){

            ocultar('buscador-avanzado');
            ocultar('etiquetas');
        });

        function ocultar(id){

            document.getElementById(id).style.display="none";
        }
        function mostrar(){

            document.getElementById('buscador-avanzado').style.display="";
            document.getElementById('etiquetas').style.display="";
        }
```

Figura 23

El archivo web_cabecera.php por su parte establece la parte visible de la cabecera definiendo el título, la selección de lenguajes disponible, la foto principal y el menú que nos permite navegar por los diversos directorios. Para la selección de lenguajes hemos empleado el método a() que crea un enlace HTML a donde le indiquemos. En las figuras 20 y 21 podemos ver el resultado final y su código respectivamente.

Figura 24. Cabecera

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
eader>
   <h1><a href="http://172.18.200.1/index">Memoria Historica</a></h1>
   <h2>    <?php echo __('Idiomas').': ';
       echo $html->a(array(
           'text' => 'Castellano',
           'href' => path('lang:es').get(),
           'class' => $config->lang == 'es' ? 'select' : null
       ));
       echo ' ';
       echo $html->a(array(
           'text' => 'Euskera',
           'href' => path('lang:eu').get(),
           'class' => $config->lang == 'eu' ? 'select' : null
       ));
       echo ' ';
       echo $html->a(array(
           'text' => 'Ingles',
           'href' => path('lang:en').get(),
           'class' => $config->lang == 'en' ? 'select' : null
       ));
       ?></h2>
   <img src="/buscador/templates/v1/img/idea.jpg" width="940" height="200" alt=""  class="imagen_cabecera"><

'header>
iav><!-- top nav -->
   <div class="menu">
       <ul>
           <li><a href="http://172.18.200.1/index"><?php echo __('Inicio') ?></a></li>
           <li><a href=/<?php echo ($config->lang);?>/noticias > <?php echo __('Noticias') ?></a></li>
           <li><a href=/<?php echo ($config->lang);?>/buscar > <?php echo __('Buscador') ?></a></li>
       </ul>
   </div>
'nav><!-- end of top nav -->
```

Figura 25. Codigo de web_cabecera.php

Después de estos dos archivos el contenido de la web varía dependiendo del directorio en el que nos encontremos. Para hacer más fácil de entender la explicación seguiremos el orden del menú (index, noticias, buscador y ficha).

El contenido del directorio index se basa en el archivo web_introducción php y es muy sencillo. Contiene un título, un par de párrafos de presentación (en castellano y en euskera) y una foto acompañándolos. Sin embargo se reserva un hueco en el margen izquierdo para añadir un pequeño buscador simple (solo se puede buscar por nombre) del cual explicaremos su funcionamiento detalladamente cuando lleguemos al directorio del buscador.

```
<section id="content"><!-- #content -->
    <article>
        <h2><?php echo __("Presentación") ?></h2>
        <img src="/buscador/templates/v1/img/Sartaguda.jpg" width="40%" alt="" class="alignright" /><p>El Parlamento de
        <p>HITZAURREA. Nafarroako Legebiltzarrak, guda zibila hasi eta Franco Jeneralaren heriotza bitartean, Nafarroan
    </article>
</section><!-- end of #content -->
<aside id="sidebar"><!-- sidebar -->

    <form action="<?php echo path('buscar'); ?>" method="get" class="background">
        <input type="text" name="nome" />
    <?php echo $html->formSubmit(' '); ?>
    <br><br><br><br>

    </form>

</aside><!-- end of sidebar -->
```

Figura 26. Código de web_introducción.php

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Figura 27. Contenido de index

Siguiendo con el orden establecido pasamos al directorio noticias. En él se encuentran las noticias colgadas por el grupo de investigación con el fin de dar mayor difusión a su trabajo y a las actividades que realizan. Las noticias tienen una estructura divida en fecha, titular, subtítulo, cuerpo de la noticia, foto y pie de foto. Todos estos campos están disponibles en la base de datos en la tabla noticias y imaxes, por lo que realizaremos una consulta a la base de datos mediante el método $data->select() en el cual podemos introducir hasta tres parámetros. En este caso nuestros parámetros son la tabla en la que buscar (noticias), los campos en los que buscar (todos los de la tabla noticias añadiendo el campo foto de la tabla imaxes para las noticias que tengan una foto disponible) y la variable sobre la que almacenaremos la consulta ($noticias).

Una vez obtenidos los datos volcamos los campos en el fichero HTML, repitiendo este proceso una vez por cada resultado obtenido, de forma que así mostramos todas las noticias que tenemos en la tabla.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

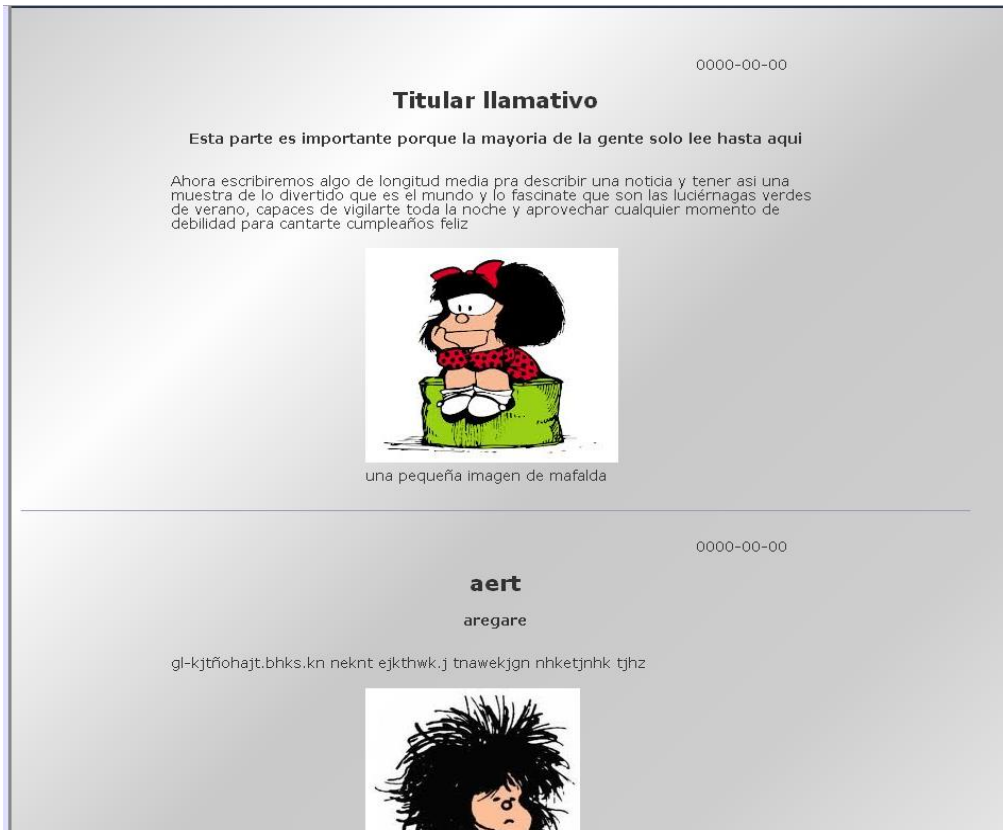


Figura 28. Noticias

```php
<?php

$data->select('noticias',
     array(
        'fields' => '*',
        'addTables' => array(
            array(
                'table' => 'imaxes',
                'fields' => array('id', 'foto')
            )
        )
    ),'noticias'
);

?>
<section noticia_global">
    <article class="articulo"><!-- #content -->
<?php
foreach ($noticias as $noti ) {
?>




    <div class="noticia">
        <p class="fecha_noticia"><?php echo $noti['fecha']; ?></p>
        <h2 align="center" > <?php echo $noti['titular']; ?></h2>
        <h1 align="center"> <?php echo $noti['subtitulo']; ?></h1><br><br>
        <p><?php echo $noti['cuerpo']; ?></p>
        <img src=<?php echo "/buscador/media/".$noti['imaxes']['foto'];  ?>  class="foto_noticia"/>
        <p class="pie_foto"><?php echo $noti['pie_de_foto']; ?></p>

    </div>


<?php } ?>
</article>
</section>
```

Figura 29. Codigo de web_noticias.php

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

El directorio buscar es el más importante de nuestra web ya que cumple la función para la que fue pensada inicialmente. En él cargaremos tres archivos: web_buscador.php, web_paginacio.php y web_listado.php.

El archivo web_paginacion.php se encarga del diseño del cajetín de búsqueda. Este cajetín está compuesto por un hueco en el que introducir un nombre para buscar, una pestaña que realiza el listado de todas las personas en la base de datos, otra pestaña que extiende el panel de búsqueda avanzada, y por último un botón con forma de lupa que manda la solicitud de búsqueda de los campos rellenados.

Para la extensión del desplegable usamos un par de funciones JavaScript, la que está definida en el archivo web_head.php que hace que el panel desplegable se muestre solo cuando se le clica en la pestaña, y otra función que está definida en el archivo javascritpt-comun.js (figura 26). Esta función se encarga de que al aparecer el panel de búsqueda avanzada desaparezca el panel de búsqueda simple.

```
(document).ready(function() {

   //Opcions de busca
   $('#opcions-busca a').click(function () {
       var li = $(this).parent();

       if ($(li).hasClass('select')) {
           $(li).removeClass('select');
           $('#buscador-avanzado').slideUp('normal');
           $('#etiquetas').slideUp('normal');
           $('#buscador-simple').slideDown('normal');
           //$('#texto-portada').fadeTo("slow", 1);
       } else {
           $(li).addClass('select');
           $('#buscador-avanzado').slideDown('normal');
           $('#etiquetas').slideDown('normal');
           $('#buscador-simple').slideUp('normal');
           //$('#texto-portada').fadeTo("slow", 0.33);
       }
       return false;
   });

   //Limpar consultas de buscas
   $('#buscador form').submit(function () {
       if ($('#buscador-simple').is(':visible')) {
           $('#buscador-avanzado').remove();
       } else {
           $('#buscador-simple').remove();
       }
   });

   //Listado de fichas
   $('#listado div.ficha').mouseover(function () {
       $(this).addClass('ficha_hover');
   });
   $('#listado div.ficha').mouseout(function () {
       $(this).removeClass('ficha_hover');
   });
});
```

Figura 30

Para realizar el panel desplegable (figura 27) los hemos separado en 2 elementos. El primero está formado por las etiquetas de las opciones de búsqueda mientras que el

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

segundo está formado por las pestañas desplegables y los huecos de búsqueda. A la hora de dar valores a las opciones desplegables utilizamos el archivo d/buscador/data/datos_buscador.php. Este archivo realiza consultas a la base de datos sobre los diferentes campos por los que se puede buscar una persona. Hay que tener en cuenta que cuando buscamos por lugar de nacimiento o residencia tenemos para elegir datos de tanto municipios como de localidades.

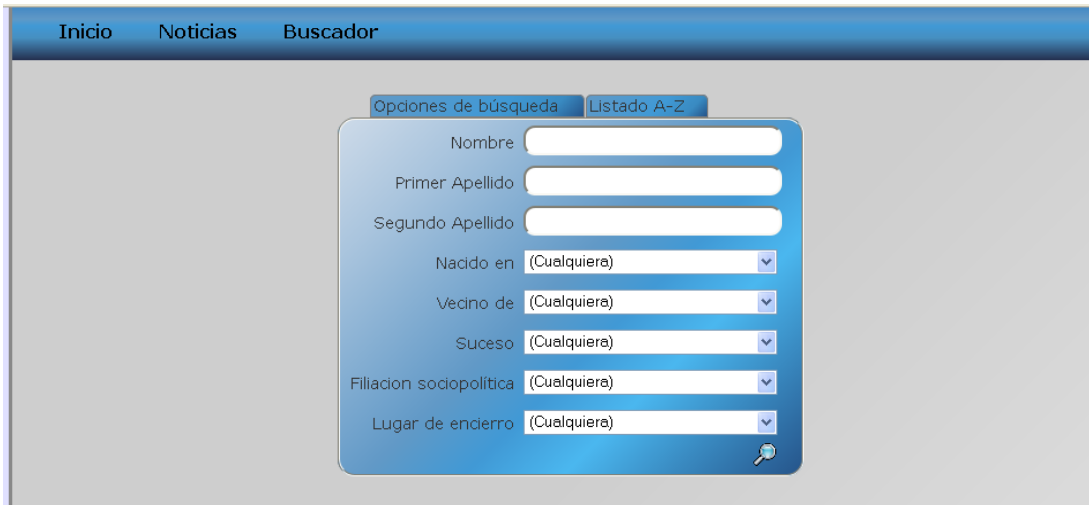


Figura 31. Panel desplegable

Una vez hemos seleccionado los datos por los cuales se va a buscar a la persona, pulsamos la lupa que envía todos estos datos al archivo buscar.php en la carpeta buscador/data/. En el archivo buscar se realizan 2 acciones principalmente. Con los datos enviados por el formulario se crea un array $preferencias que contiene todos los parámetros que debe seguir la consulta a la base de datos. Este array lo añadimos al array $query mediante el método $Search->like($preferencias). Una vez hecho esto, al array query le añadimos los campos que deseamos obtener en nuestra consulta. Por último realizamos la consulta select(persoas,$query,resultado), de la cual guardamos en la variable $resultado los datos que nos devuelve. El proceso de creación de la variable $query se puede observar en las figuras 28 y 29.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
$preferencias = array(
    'table' => 'personas',
    'fields' => array('nome', 'sexo', 'apellido1','apellido2', 'apodo', 'traxectoria'),
    'vars' => array('nome', 'apellido1','apellido2', 'natural', 'vecinanza', 'filiacion', 'suceso', 'encierro'),
    'text' => stripslashes($vars->str('buscar')),

    'field_settings' => array(

        'natural' => array(
            'table' => $tabla,
            'name' => 'natural_'.$nombre,
            'field' => 'nombre',
            'id_search' => true,
            'id_to_string' => $busqueda,
            'id_to_string-id_val' => 'id',
            'id_to_string-string_val' => 'nombre',
        ),

        'vecinanza' => array(
            'table' => $tabla2,
            'name' => 'vecino_'.$nombre2,
            'field' => 'nombre',
            'id_search' => true,
            'id_to_string' => $busqueda2,
            'id_to_string-id_val' => 'id',
            'id_to_string-string_val' => 'nombre',
        ),


        'suceso' => array(
            'table' => 'sucesos',
            'field' => 'nombre',
            'id_search' => true,
            'id_to_string' => $sucesos,
            'id_to_string-id_val' => 'id',
            'id_to_string-string_val' => 'nombre',
        ),
        'filiacion' => array(
            'table' => 'fil_socpol_sindical',
            'field' => 'nombre',
            'id_search' => true,
            'id_to_string' => $filiacion,
            'id_to_string-id_val' => 'id',
            'id_to_string-string_val' => 'nome',
```

Figura 32. Preferencias

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
$query += array(
    'fields' => array('id', 'nome', 'apellido1','apellido2',  'apodo', 'sexo','anio_nacimiento','mes_nacimiento','dia_nacimiento'),
    'conditions' => array(
        'validado' => 1
    ),
    'addTables' => array(
/*
        array(
            'table' => 'filiacions',
            'fields' => array('id', 'nome'),
        ),
*/

        array(
            'table' => 'municipios',
            'relation_name' => 'vecino_municipio',
            'fields' => array('id', 'nombre'),
            // 'addTables' => array(
                // array(
                    // 'table'=>'provincias',
                    // 'fields'=>array('id','nombre')
                // )
            // )
        ),
        array(
            'table' => 'municipios',
            'relation_name' => 'natural_municipio',
            'fields' => array('id', 'nombre'),
            // 'addTables' => array(
                // array(
                    // 'table'=>'provincias',
                    // 'fields'=>array('id','nombre')
                // )
            // )
        ),
        array(
            'table' => 'localidades',
            'relation_name' => 'vecino_localidad',
            'fields' => array('id', 'nombre'),
            // 'addTables' => array(
                // array(
                // 'table'=>'provincias',
                // 'fields'=>array('id','nombre')
                // )
            // )
```

Figura 33

Una vez obtenidos los datos de nuestra consulta obtendremos una variable $resultado con toda la información que necesitamos para completar el listado de personas del directorio buscar. Para ello, al final del archivo web_buscador.php se carga el template "contenido_extendido" el cual devuelve el archivo web_listado.php cuando estamos en el directorio buscar. A la hora de realizar el listado tenemos que tener en cuenta varias cuestiones: tiene que ser sencillo de manejar, tiene que mostrar todos las personas que coincidan con la búsqueda y tiene ofrecer algo de información general de la búsqueda (cuantos resultados se han obtenido o en que página nos encontramos del total de posibles). Para cumplir con esto, lo primero que tenemos que hacer es obtener el número de resultados totales. Para ello utilizamos un atributo interno de la clase data que cuando realiza una consulta obtiene el número de resultados, el número total de páginas (cada página puede tener hasta 50 resultados), la página en la que nos encontramos, la última página del total, y la primera y la última página del rango en la que nos encontremos (nosotros lo hemos definido para que abarque un rango de 4 páginas, donde la página nuestra actual es la tercera, teniendo 2 páginas previas a nosotros y 1 página seguida de la nuestra).

Una vez hemos mostrado el número de resultados llamamos a el archivo web_paginacion.php que se encargará de construir la cabecera que nos permite navegar

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

entre las páginas. La cabecera para la navegación tiene 3 elementos. El primero de ellos es una pestaña de selección que permite ordenar el resultado de la búsqueda por nombre, apellido y fecha de nacimiento. El segundo de ellos nos muestra en página sobre el total nos encontramos. El último elemento es una serie de botones que nos permiten desplazarnos a la primera página, a la anterior a la nuestra, a la siguiente, a la última, o a cualquier página dentro de nuestro rango (por ejemplo si nos encontramos en la página 20 podremos seleccionar las páginas 18, 19, 20 y 21).

```php
<?php defined('ANS') or die(); ?>

<div class="orden_paginacion">
    <form class="lista_ordenacion">
        <div class="inline">
            <?php
            echo $html->formSelect(
                array(
                    'nome' => __('Nome'),
                    'apellido1' => __('Apelidos'),
                    'anio_nacimiento' => __('Nacimiento'),
                ),
                array(
                    'variable' => 'orde',
                    'autosubmit' => true,
                ),
                __('Ordenar por')
            );

            //$get = arrayGet('', array('nome', 'sexo', 'natural', 'vecinanza', 'filiacion', 'suceso', 'morte', 'buscar'));
            $get = arrayGet('', array('nome', 'sexo', 'natural', 'vecinanza', 'suceso', 'morte', 'buscar'));
            foreach ($get as $k => $v) {
                echo $html->formHidden($v, $k);
            }
```

Figura 34. Código de web_paginación (ordenar listado)

```php
<p class="pagina_actual"><?php __e('Páxina %d de %d', $pagination['page'], $pagination['total_pages']); ?></p>

<ul class="numeracion">
    <li class="primera"><?php echo $pagination['page'] == 1 ? '<span>'.__('Primeira').'</span>' : $html->a(__('Primeira'), path().get('pax',
    <li class="anterior"><?php echo $pagination['previous'] ? $html->a(__('Anterior'), path().get('pax', $pagination['previous'])) : '<span>'

    <?php for ($n = $pagination['first']; $n <= $pagination['last']; $n++) { ?>
    <li<?php echo ($n == $pagination['page']) ? ' class="select"' : ''; ?>>
        <?php echo $html->a($n, path().get('pax', $n)); ?>
    </li>
    <?php } ?>

    <li class="Siguiente"><?php echo $pagination['next'] ? $html->a(__('Seguinte'), path().get('pax', $pagination['next'])) : '<span>'.__('Se
    <li class="Ultima"><?php echo $pagination['page'] >= $pagination['total_pages'] ? '<span>'.__('Última').'</span>' : $html->a(__('Última')
</ul>
div>
```

Figura 35. Código web_paginación (desplazamiento por las páginas)

Figura 36. Cabecera del listado

Ahora que ya hemos establecido la cabecera seguimos con el archivo web_listado.php para crear el listado. Para ello solamente tenemos que imprimir los

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

campos que nos interesen de cada registro de la variable $resultado (por ejemplo nombre, apellidos o lugar de nacimiento). Repetiremos este proceso un máximo de 50 veces (es el número máximo de resultados que admite una página), dejando el resto de registros pendientes para las demás páginas del listado. Además, el elemento div donde se coloca cada ficha individual (class=”ficha_lista2”) se ha configurado para que se pueda clicar en él, acción que nos conducirá a la ficha detallada de la persona elegida.

```
v id="listado" class="lista">

<?php foreach ($resultado as $persona) { ?>
<div class="ficha_lista">
    <div class="ficha_lista2" onclick="location.href='<?php echo path('ficha', $persona['id'])?>';">
        <div class="identificacion">

            <h2><?php echo $persona['nome'].' '.($persona['apodo'] ? ' "'.$persona['apodo'].'"' : ''); ?><br>
                <?php echo $persona['apellido1'].' '.$persona['apellido2']?>

            </h2>


             <?php
            if ($persona['imaxes']) {
                echo "<img src=\"/buscador/templates/v1/img/camara-icono.png\"  class=\"sonido\" title=\"".__("Esta persona tiene fotos")."\"/>"
            }
            if ($persona['entrevistas']) {
                echo "<img src=\"/buscador/templates/v1/img/sonido-icono.png\"  class=\"sonido\" title=\"".__("Archivo de sonido disponible")."\

            }
            ?>

        </div>

        <div class="detalles">
            <p><?php if ($persona['sexo']=='Hombre') __e($persona['sexo']);
                    else if ($persona['sexo']=='Mujer') __e($persona['sexo']);
             ?>
               <br><?php echo __('Nacido en ').($persona['natural_localidad']['nombre'] ? $persona['natural_localidad']['nombre'].' , '.$persona['n
                <?php echo ($persona['natural_provincia']['nombre']  ? " (".$persoa['natural_provincia']['nombre'].')' : '')?>
                <?php echo __('en ').$persona['anio_nacimiento'].'-'.$persona['mes_nacimiento'].'-'.$persona['dia_nacimiento']?>
               <br><?php echo __('Vecino de ').($persona['vecino_localidad']['nombre'] ? $persona['vecino_localidad']['nombre'].' ('.$persona['veci
               <?php echo ($persona['vecino_provincia']['nombre']  ? ' ('.$persona['vecino_provincia']['nombre'].')' : '')?>

            </p>
            <p class="t1">
                <?php echo $persona['traxectoria']; ?>
            </p>
        </div>
    </div>
</div>

<?php ) ?>
```

Figura 37. Código de web_listado .php

En la figura 34 podemos ver una pequeña muestra del listado. El elemento coloreado es aquel sobre el que tenemos posado el cursor (los colores y estilos se explicaran más adelante).

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

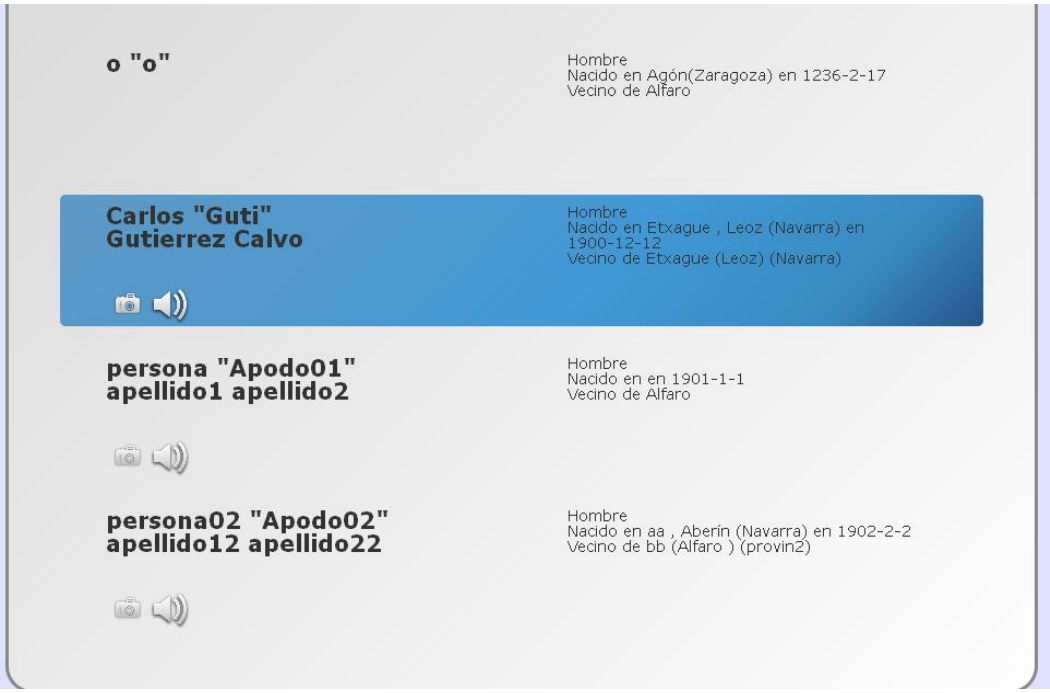


Figura 38. Listado

Como hemos mencionado antes, clicar en cualquier persona del listado nos llevará a su ficha detallada. El objetivo de esta ficha es profundizar en todos los detalles conocidos esta persona. Para lograr mostrar todos los detalles necesitamos 3 archivos: el archivo functions.php localizado en buscador/includes/, el archivo ficha.php que se encuentra en la carpeta buscador/data/ y por último el archivo web_ficha.php que tenemos en la carpeta de los templates. El archivo ficha.php se encarga de obtener información de esta persona mediante consultas a la base de datos, el archivo functions.php se encarga de procesar parte de esa información y la prepara para mostrarla en la web, por último el archivo web_ficha.php crea la ficha final detallada de la persona.

Al hacer las consultas en la base de datos nos interesa obtener varias cosas:

- Todos los datos de la persona que muestren información relevante sobre su vida, como dónde y cuándo nació, sus padres, su mujer, cuántos hijos tiene, a que se dedicaba, sus afiliaciones políticas etc.
- Los sucesos padecidos por esta persona (si fue detenido, si recibió algún tipo de castigo o si fue ejecutado por ejemplo).
- Los lugares donde ha estado encerrada la persona, el delito por el cual se le encerró y el periodo que duró dicho encarcelamiento.
- Las imágenes y entrevistas en las que aparezca esta persona.
- Las fuentes de las cuales se ha obtenido información de la persona.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Lo comentado anteriormente da una idea de las consultas que vamos a llevar a cabo en la base de datos. El grueso de estas consultas se realizará en las tablas personas, registro de cautiverio y sucesos. Para enfocar estas consultas sobre la única persona que nos interesa en este momento se utilizará la variable $vars->int(0) que en el momento de clicar sobre la ficha de una persona almacena su id.

```php
<?php
defined('ANS') or die();

if ($config->lang!='en'){
    $lengua=$config->lang;
}
else $lengua='es';
$data->select('persoas', array(
    'fields' => '*',
    'conditions' => array(
        'id' => $vars->int(0),
        'validado' => 1,
    ),
    'addTables' => array(

        array(
            'table' => 'fil_socpol_sindical',
            'fields' => array('id', 'nombre'),
        ),

        array(
            'table' => 'municipios',
            'relation_name' => 'vecino_municipio',
            'fields' => array('id', 'nombre'),

        ),
        array(
            'table' => 'municipios',
            'relation_name' => 'natural_municipio',
            'fields' => array('id', 'nombre'),

        ),
        array(
            'table' => 'localidades',
            'relation_name' => 'vecino_localidad',
            'fields' => array('id', 'nombre'),

        ),
        array(
            'table' => 'localidades',
            'relation_name' => 'natural_localidad',
            'fields' => array('id', 'nombre'),


        ),
```

Figura 39. Una pequeña muestra de la consulta sobre la tabla persoas realizada en ficha.php

Una vez obtenida toda la información procedemos a crear la ficha para mostrarla. Para ello en el archivo web_ficha.php hemos diseñado una estructura que como explicamos en la fase de diseño consiste en la siguiente estructura (figura 36).

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

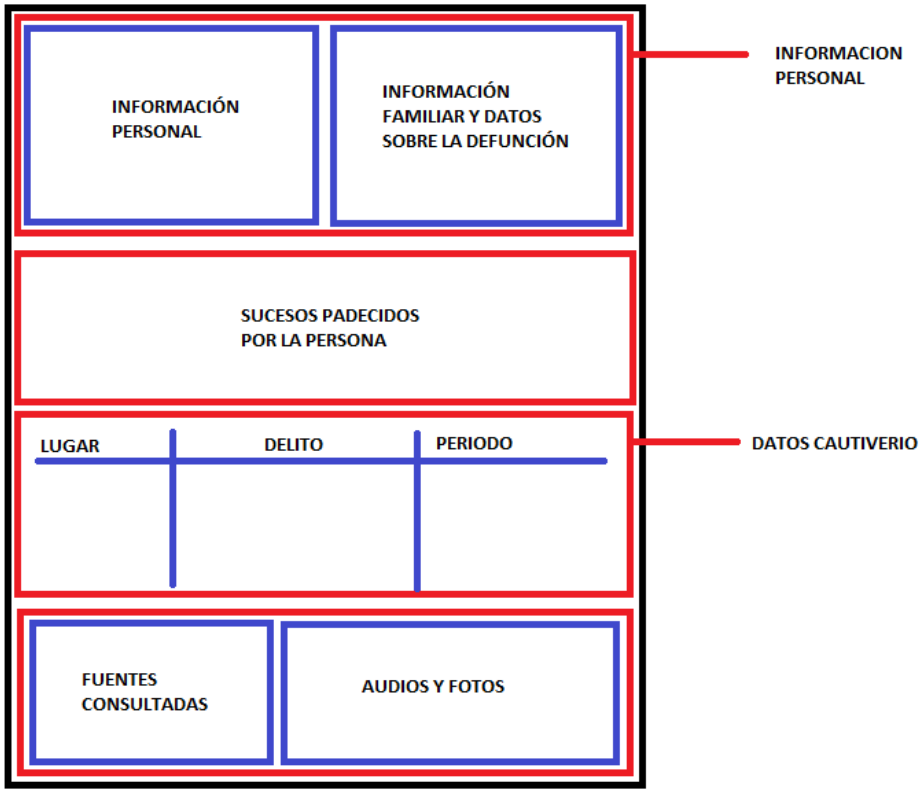


Figura 40. Estructura de la ficha personal

Para la parte con información personal hemos utilizado los datos resultantes de la consulta a la tabla de persoas. Estos datos estaban almacenados en la variable $ficha de la cual los hemos extraído tanto de forma directa como utilizando las funciones del archivo functions.php. En este archivo tenemos 3 funciones: cotidiana(), familia() y descripción(). La función cotidiana la hemos utilizado para extraer datos del lugar y fecha de nacimiento, lugar de residencia, profesión y afiliaciones políticas. La función familia la hemos utilizado para obtener los datos familiares de la persona como con quién estaba casado, sus padres y el número de hijos que tuvo. Por último la función descripción se encarga de obtener los datos de sobre la muerte (en caso de haberse producido) de la persona. Estas 3 funciones no solo recopilan datos sino que también les dan forma mediante párrafos y saltos de línea.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```php
<?php defined('ANS') or die(); ?>


    <div class="ficha_individual">
        <div class="ficha_desc">
            <h2><?php
                echo $ficha['nome'];
                if ($ficha['apodo']) {
                    echo ' "'.$ficha['apodo'].'"';
                }
                echo '<br>'.$ficha['apellido1'].' '.$ficha['apellido2'];
            ?></h2>
            <p class="t1"><?php echo cotidiana($ficha); ?></p><br><br>
            <p class="t1"><?php echo $ficha['trayectoria']; ?></p>


        </div>
        <div class="ficha_desc_derecha">
            <?php if ($ficha['padre']||$ficha['madre']||$ficha['conyuge']||$ficha['hijos']){ ?>
                <h1><?php echo __("Familia"); ?> </h1>
            <?php } ?>
            <p class="t1"><?php echo familia($ficha); ?></p>
            <?php if ($ficha['morto']){ ?>
                <h1> <?php echo __("Informe de defunción"); ?> </h1>
            <?php } ?>
            <p class="t1"><?php echo descricion($ficha); ?></p>
        </div>
```

Figura 41. Código de web_ficha referente a los datos personales

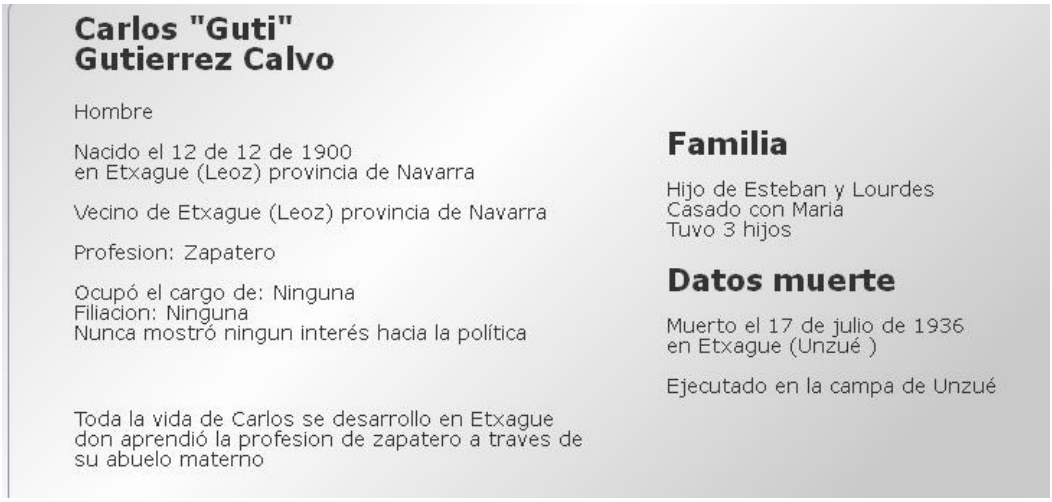


Figura 42. Datos personales de la ficha

Las zonas de la ficha referidas a los sucesos y a los registros de cautiverio siguen un procedimiento similar. En las consultas a la base de datos se obtiene toda la información

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

de todos estos hechos. Después cada suceso (o registro de cautiverio) se va volcando en la ficha para mostrar los datos correspondientes. Pero estas dos zonas difieren a la hora de darles forma. Los sucesos se muestra como pequeñas fichas con una breve descripción del mismo cada uno, mientras que los registros de cautiverio se muestran en forma de tabla en la cual cada fila pertenece a un registro diferente y las tres columnas que tiene se corresponden con el lugar de cautiverio, el delito cometido y el periodo del cautiverio.

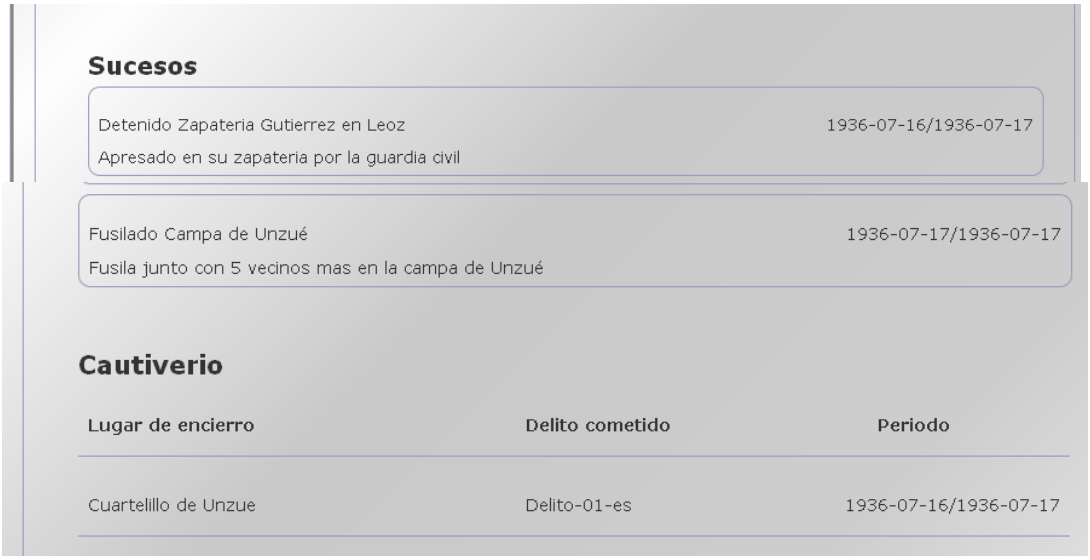


Figura 43 . Sucesos y registro de cautiverio de la ficha.

Por último nos queda la parte en la que mostramos las referencias y la parte en la que mostramos las fotos y los audios de los que disponemos. Para la zona de las referencias nos limitamos a listar los resultados de las consultas de la base de datos de las tablas que sirven para almacenar las fuentes (registro de cautiverio o fuentes bibliográficas por ejemplo). Para las imágenes utilizaremos un pequeño javascript que pasa el tamaño de muestra de la imagen a su tamaño real cuando hacemos click en ella. Este fragmento de código está ubicado en el archivo modulo-imaxes.ph de la carpeta templates. El archivo de audio depende de otro archivo de la carpeta templates llamado modulo.audios.php, en el cargamos el javascript flowplayer que permite la reproducción de archivos de sonido mp3, además de otorgarle una pequeña interfaz para hacer posible la escucha.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Estilos

Una vez construida la estructura HTML de la web es hora de darle estilos. Para ello hemos creado cuatro archivos styles.css, stylesnoticia.css, stylesbuscador.css y styleficha.css. Estas hojas de estilos serán las encargadas de que darle el color y la forma adecuadas a nuestra web y se encuentran en la carpeta buscador/templates/v1/css/, además para que podamos utilizarlas han de ser cargadas en la variable templates, de forma que al declarar la cabecera de nuestra página con el archivo web_head.php se indique que utilizamos estos archivos. A continuación explicaremos las funciones principales de cada hoja.

La hoja styles.css se encarga de las características que afectan a toda la página (como los colores) pero también se encargan del directorio index. Las medidas generales más importantes que hemos implementado están relacionadas con los enlaces. Hemos modificado los estilos de los mismos para que a simple vista no se distinga de un texto normal, pero que al pasar el ratón por encima adquieran un sombreado ligero y aumente ligeramente el tamaño de la fuente. También les hemos quitado los efectos que se les añadían una vez los había clicado o cuando están seleccionados, la excepción a esto se da en las elecciones de idioma de la cabecera, en la que al seleccionarlos se les rodea con un borde de rayas entrecortadas. A todo esto hay que añadir que la mayoría de los elementos de la web tienen los bordes redondeados para darle un aspecto más elegante a nuestro portal web y que hemos utilizado las instrucciones padding y margin para dejar todos los elementos de la web de la forma más ordenada posible y sin que interfieran entre ellos.

A la hora de diseñar el color decidimos que queríamos tonalidades cercanas al blanco en la mayoría de la página, y un color azul para resaltar elementos puntuales de la misma. Partiendo de este punto aplicamos un color gris claro para el fondo sin contenido de la web. Para distinguir el contenido de la web del fondo lo rodeamos de un pequeño borde y le dimos un color de gris un poco más oscuro y con degradados (que a veces le dan un ligero aspecto metalizado). Para los elementos como la cabecera, el menú del navegador o el buscador avanzado utilizamos degradados sobre tonos azules, estos son más oscuros en los dos primeros que en el buscador.

```
background: rgb(206,219,233); /* Old browsers */
background: -moz-linear-gradient(-45deg,  rgba(206,219,233,1) 0%, rgba(170,197,222,1) 17%, rgba(97,153,199,1) 47%,
background: -webkit-gradient(linear, left top, right bottom, color-stop(0%,rgba(206,219,233,1)), color-stop(17%,rgb
background: -webkit-linear-gradient(-45deg,  rgba(206,219,233,1) 0%,rgba(170,197,222,1) 17%,rgba(97,153,199,1) 47%,
background: -o-linear-gradient(-45deg,  rgba(206,219,233,1) 0%,rgba(170,197,222,1) 17%,rgba(97,153,199,1) 47%,rgba(
background: -ms-linear-gradient(-45deg,  rgba(206,219,233,1) 0%,rgba(170,197,222,1) 17%,rgba(97,153,199,1) 47%,rgba
background: linear-gradient(135deg,  rgba(206,219,233,1) 0%,rgba(170,197,222,1) 17%,rgba(97,153,199,1) 47%,rgba(65,
filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#cedbe9', endColorstr='#26558b',GradientType=1 )
```

Figura 44. Ejemplo de la implementación del degradado

El diseño de las noticias no tiene mucha complejidad, se limita centrar los textos tanto del titular como del subtítulo y el cuerpo de la noticia y separar las noticias entre sí mediante un fino borde inferior (1px) de un color gris azulado. Este tipo de borde (de 1

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

pixel de grosor y de color gris azulado) lo seguiremos usando en el resto de la web para separar componentes.

El diseño del buscador y el listado es un poco más elaborado. El buscador avanzado cumple con las descripciones generales que hemos dado antes (degradado de azul y elementos redondeados). El listado de personas lo diferenciamos en dos zonas: el encabezamiento y el listado en sí. El encabezamiento está separado de la lista mediante el mismo borde fino que hemos hablado antes. Los elementos que permiten navegar por las páginas del listado también están rodeados del mismo borde y también tienen contornos redondeados. La lista de personas sigue el mismo estilo que estamos utilizando para los demás elementos: cada ficha personal está separada del resto por un margen y rodeada con el borde habitual. La peculiaridad del listado llega cuando pasamos el ratón por encima de alguna ficha, ya que como hemos mencionado antes, al clicar sobre una nos llevara a la ficha detallada de la persona. Por lo tanto, y para apreciar bien que ficha se está seleccionando le daremos un color azul a la ficha sobre la que pasemos el cursor del ratón.

Figura 45. Diferenciación de la ficha mediante el color

Por último en la ficha detallada hemos aplicado los mismos estilos que hemos estado explicando, separando las diversas zonas de la ficha mediante márgenes y utilizando los bordes para el listado de sucesos y la tabla de registro de cautiverio.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Traducciones

A la hora de preparar las traducciones hemos utilizado dos funciones de phpcan, la función __() y la función __e(). La primera de estas funciones tiene la propiedad de devolver la cadena de texto que se le pasa como argumento en la lengua en la que corresponda. La función __e() funciona de igual forma que la anterior pero mientras que la anterior devuelve la cadena resultante como resultado de la función, la función __e() la imprime directamente.

Pero obviamente estás funciones no traducen sin ninguna ayuda las cadenas que le pasemos. Para ello tenemos que crear un fichero .po de cada lengua. Estos ficheros contienen todas las líneas originales que deben traducirse y a continuación de estas su traducción en el idioma correspondiente como se puede ver en la figura 42.

```
#: phpcan/includes/class_data.php:78
#, php-format
msgid "The variable %s has been previously registered"
msgstr "%s aldagaia lehenago erregistratu da"

#: phpcan/includes/class_data.php:126
#, php-format
msgid "The table %s doesn't exists"
msgstr "%s taula ez da existitzen"

#: phpcan/includes/class_data.php:130
#, php-format
msgid "The field %s doesn't exists"
msgstr "%s eremua ez da existitzen"

#: phpcan/includes/class_search.php:60
#, php-format
msgid "The mode \"%s\" does not exists"
msgstr "\"%s\" modua ez da existitzen"

#: phpcan/includes/class_search.php:354
#, php-format
msgid "This function does not exists (%s)"
msgstr "Funtzio hau ez da existitzen(%s)"

#: phpcan/includes/class_image.php:20
msgid "No image library is loaded"
msgstr "ez da argazki liburutegia kargatu"
```

Figura 46. Traducciones en el archivo eu.po

Para realizar las traducciones utilizaremos el programa poedit que permite ir recorriendo línea a línea mientras se añade la traducción además de indicar los posibles errores que se puedan cometer. Al guardar un archivo .po desde poedit creamos un archivo .mo que es el que se utiliza para que las funciones __() y __e() realicen la traducción.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Volcado de la base de datos

Los miembros del grupo de investigación guardan todos sus datos en archivos excel. Para convertir estos archivos Excel en información en las tablas de mysql tenemos que seguir un pequeño proceso.

Lo primero es asegurarnos que los archivos Excel tienen todos los campos necesarios y en el orden correcto, la única salvedad se pude dar con el campo id presente en todas la tablas, ya que si no está podemos añadirlo mediante el programa encode de awk. También hay que añadir que si se da el caso el que algún campo nos lo dan con nombre en vez de números (puede darse que nos den el nombre del campo relacionado en vez de su id) podemos intentar la sustitución de los nombres por números con el archivo transformador.flex aunque el éxito de esta operación no es nada seguro, así que lo mejor es que nos pasen el archivo correctamente.

Una vez nos hemos asegurado que los campos están correctamente subiremos el archivo a nuestro servidor a la carpeta buscador/bbdd. Hecho esto, utilizando el programa putty nos conectamos a nuestro servidor para comprobar la codificación de estos ficheros mediante la terminal. Ejecutando el comando file 'nombre_fichero' obtenemos la codificación del mismo. Si esta codificación es utf-8 lo dejamos como está, y si no la cambiamos mediante el comando:

iconv -f 'codificación actual' -t UTF-8 'nombre del fichero' > 'nombre del fichero-utf8.csv'

Este comando crea un fichero codificado en utf-8 a partir del fichero original.

Llegados a este punto solo nos falta cargar el fichero en nuestra base de datos, para ello nos conectamos desde la terminal del programa putty a la base de datos usando el comando: mysql –u 'nombre de usuario' –p 'contraseña'. Una vez conectado utilizamos el comando:

load data local infile 'ruta+nombre del archivo' into table 'nombre de la tabla' fields terminated by ';' lines terminated by '\n';

Si se ha seguido todos los pasos se conseguirá volcar el archivo en la tabla.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Utilización del módulo de administración

Aunque como ya hemos dicho este módulo es el mismo que el del proyecto gallego, es necesario mencionar algunas pequeñas modificaciones que se le realizaron, así como su funcionamiento a la hora de gestionar las tablas y actualizar la base de datos.

Las modificaciones correspondientes al módulo de administración se limitan a la alteración del interfaz de las tablas para adecuarlo a las modificaciones de la base de datos. Estas modificaciones se realizan en el fichero admin_tables.php que se encuentra en buscador/config. En este fichero se almacena información de las tablas de cara a su representación en la interfaz de administración. Esta información consiste en:

- La clasificación de la tabla en contenido o texto.
- Las acciones que se pueden realizar con dicha tabla (listarla, crear nuevos registros o exportarla en formato csv)
- Las relaciones de nuestra tabla con otras tablas. Ya hemos mencionado que las tablas almacenan el campo id de algunas tablas. Definiendo correctamente la relación entre dos tablas podemos establecer que el campo perteneciente a otra tabla no se visualice por su id si no por un campo asociado a este (normalmente se utiliza el nombre). Por ejemplo la tabla registro de cautiverio tienen un campo procedente de la tabla delitos llamado id_delitos. Al establecer convenientemente la relación en este archivo, podemos hacer que en la lista seleccionable de delitos que tenemos a la hora de rellenar este campo, nos aparezca en función del nombre del delito y no de su id.
- Los campos que vamos a mostrar en la interfaz al administrador.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

```
            array(
                'table' => 'tipos_lugares_encierro',
                'field' => 'nombre',
            ),
            array(
                'table' => 'lugares',
                'field' => 'nombre',
                'name' => 'lugar',
            )
        ),
        'menu-table' => array('list-db', 'new-db', 'export-csv'),
        'menu-row' => array(
            'edit-db' => array(
                'order' => array(
                    'datos_personales' => array(
                        'nombre',
                        'sexo',
                        'sexos',
                        'anio_nacimiento',
                        'mes_nacimiento',
                        'dia_nacimiento',
                        'edad',
                        'profesions',
                        'natural',
                        'comarcas_geograficas',
                        'provincias',
                        'ambitos',
                        'lugar',
                        'vecino_municipio',
                        'vecino_comarca_geografica',
                        'vecino_provincia',
                        'vecino_ambito',
                        'fil_socpol_sindical',
                        'cargo_politico_institucional',
                    ),
                    'datos_lugar_encierro' => array(
                        'tipos_lugares_encierro',
                        'nombre_lugar_encierro',
                        'lugar',
```

Figura 47. Extracto de la configuración de admin_table

Las interfaces de las tablas en el módulo de administración tienen un manejo muy sencillo. En el interfaz donde aparecen todas las tablas (separadas en tablas de contenido, figura 44, o de texto) podemos seleccionar tres opciones: listar la tabla, crear un nuevo registro en la tabla o exportarla a formato csv. Para hacer esto último solo tenemos que clicar en esta opción y nos aparecerá un mensaje para descargar o abrir el archivo csv.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Figura 48

Si clicamos en la opción nueva, iremos a una página en la que nos aparecen todos los campos disponibles para ser rellenados. Una vez allí solamente tenemos que completar los campos vacíos y dale a guardar.

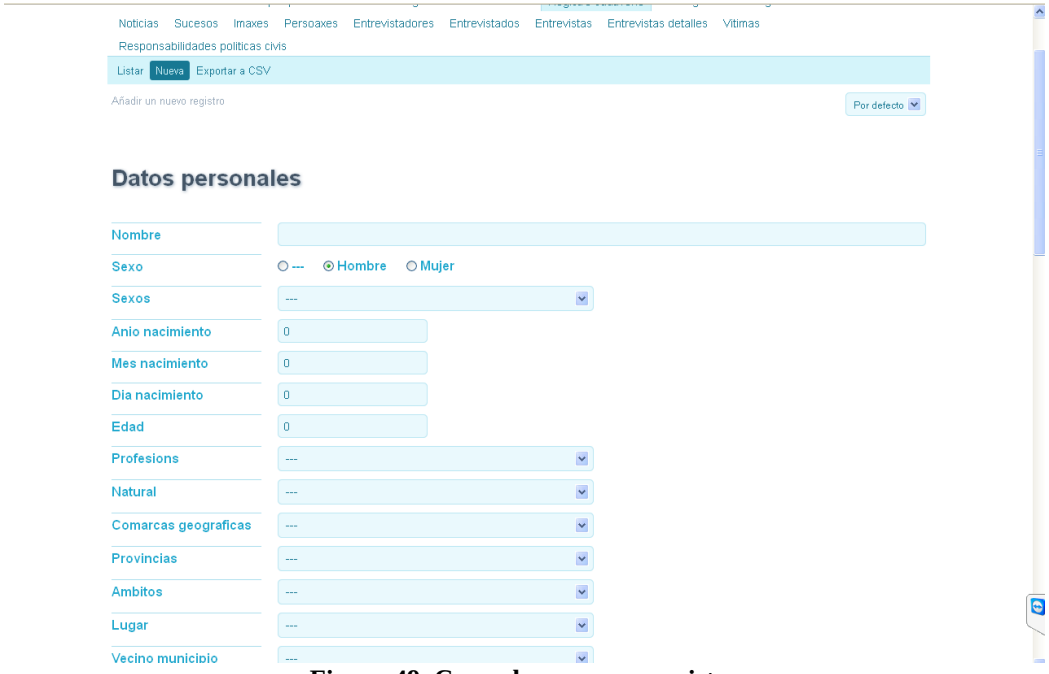


Figura 49. Creando un nuevo registro

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

En la opción de listar nos aparecerá un listado de todos los registros de la tabla, el número de registros que aparecerán en pantalla es limitado pero podemos navegar a través de las diferentes páginas que engloban la lista a través de varios botones que nos permiten desplazarnos por las páginas a voluntad. A la derecha de cada registro del listado aparece la opción editar, si clicamos nos llevará a un interfaz como el que aparece si creas un registro nuevo, solo que esta vez los campos ya tienen valores que podemos modificar. En este apartado también podemos relacionar nuestro registro de la tabla con registros de otras tablas. Para ello disponemos de un listado en la parte inferior con todas las tablas relacionables. Una vez clicamos en la tabla con la cual queremos relacionarla nos aparece la opción de listar los registros, una vez hecho el listado escogemos el registro que queremos relacionar con el nuestro y pulsamos el botón 'relacionar' que se encuentra a la derecha de esta.

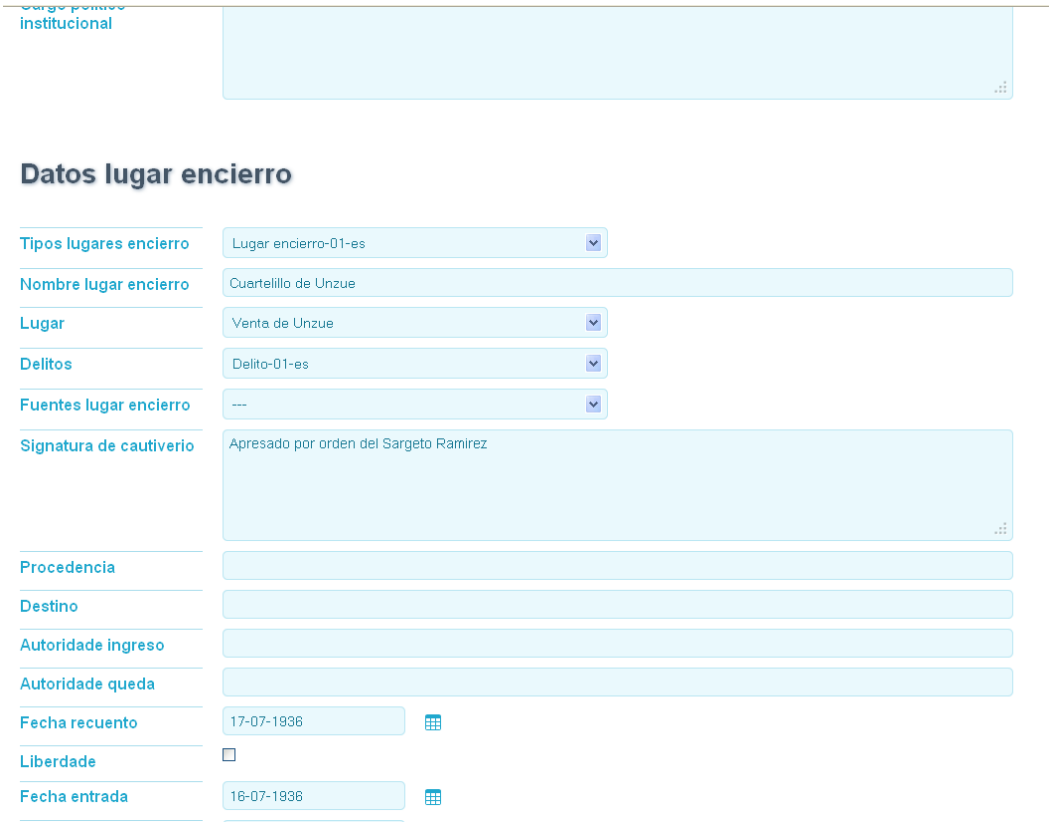


Figura 50

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

Figura 51. Opciones para relacionar tablas

Otra funcionalidad muy recurrida del módulo de administrador es la opción de actualizar la base de datos. Esta función capta los cambios hechos en la definición de las tablas del archivo tables.php e indica en rojo si hay cambios sin actualizar. Si tenemos cambios sin actualizar solamente tendremos que pulsar el botón lanzar actualización.

Figura 52. Actualización de la base de datos

Por último, cuando creamos nuevas tablas tenemos que otorgar permisos para que los demás usuario puedan interactuar con ellas. Esto se hace desde la pestaña avanzado

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

eligiendo la opción usuarios. Aquí te salen una pequeña lista de usuarios y si seleccionas cualquiera de ellos te salen sus permisos para interactuar con las tablas. Para otorgar o quitar permisos solamente tenemos que clicar en los recuadros que aparecen.

Figura 53

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CONCLUSIONES Y LÍNEAS FUTURAS

A la conclusión de este proyecto dispondremos de un portal web totalmente funcional que será puesto en manos del Servicio Informático de la UPNA para su mantenimiento y posteriores ampliaciones.

Antes de dar por finalizado este proyecto me parece importante señalar el impacto que ha tenido el grupo de investigación en el mismo. Es cierto que la experiencia de trabajar para alguien es novedosa y enriquecedora, ya que durante la carrera no se tiene ese tipo de experiencias. Aun así, me gustaría señalar que si bien la relación con los investigadores ha sido excelente, la planificación de los objetivos comunes ha sido mejorable. En particular, algunos cambios de última hora o el envío de archivos que nunca llegaron al desarrollador, así como dejar para adelante temas en los que nunca se llegó a profundizar lo necesario, dieron la sensación que muchas veces el cliente desconocía hacia donde desarrollar el proyecto. Obviamente, el desarrollador debería haber insistido más en los puntos a los que creía que no se les estaba prestando la atención necesaria.

Por otro lado, el aprendizaje de un nuevo lenguaje de programación siempre supone un reto para cualquier desarrollador, en especial cuando hay tan poco material de apoyo disponible. Sin embargo, como cuando se supera cualquier dificultad la sensación es plenamente satisfactoria.

Para finalizar, indicar las líneas futuras que puede seguir el proyecto. Lo primero sería hacer una revisión de la base de datos y establecer en la medida de lo posible los campos y tablas definitivos. Para ello también se debería repasar los campos de la parte pública de la web que se deseen traducir, ya que la traducción al euskera e inglés no alcanza a toda la web debido a que no se ha determinado con exactitud los campos que deberían ser trilingües.

Una vez repasados estos conceptos lo principal sería dotar al módulo de administración con mejores herramientas para realizar la investigación. Para ello se podría optar por añadir una herramienta que convierta los resultados de las consultas avanzadas en gráficas. Para apoyar esto debería potenciarse el sistema de búsqueda avanzada para permitir el almacenamiento y ejecución de instrucciones SQL fijadas de antemano, con el fin de facilitar la obtención de información de la base de datos.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

BIBLIOGRAFÍA

[1] Web oficial de php: http://php.net/

[2] Web con documentación acerca de PHP, HTML, CSS y JavaScript: http://www.w3schools.com/

[3] Web con la documentación de phpCan: http://idc.anavallasuiza.com/project/phpcan/

[4] Web del proyecto gallego original: http://www.nomesevoces.net/

[5] Web oficial de MySQL: http://www.mysql.com/

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

SISTEMA DE INFORMACIÓN DE REPRESALIADOS DE LA GUERRA CIVIL EN NAVARRA

Iosu Abaurrea Roldan

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

PUNTO DE PARTIDA

- Proyecto basado en el proyecto gallego “Nomes e Voces”.
  - Creado en 2006 a traves de un convenio entre universidades gallegas y la Consejería de Cultura de la Xunta de Galicia
  - Objetivo:
    - Estudio de la represión durante la guerra civil en Galicia.
    - Recopilación de información en una base de datos.
    - Mostrar al público información de las víctimas de la guerra civil.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

PUNTO DE PARTIDA

- Grupo de investigación “Fondo Documental de la Memoria Histórica en Navarra”
  - Formado por Emilio Majuelo, Juan Carlos García, Gotxon Garmendia y Fernando Mendiola.
- Intenta trasladar la idea original a el ámbito de Navarra y a un periodo mas largo.
- Se encarga el Servicio Informático de la UPNA.
  - Se aparta debido a la escasez de recursos personales.
  - Se ofrece como Proyecto de Fin de Carrera

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

PUNTO DE PARTIDA

- Se utiliza como base el código del proyecto gallego.
  - Basado en phpCan
  - También utiliza HTML, PHP y JavaScript para la creación de la web.
  - La base de datos se gestiona mediante MySQL.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

OBJETIVOS

- Web divida en un módulo público y un módulo privado
- Modulo público
  - Permite consultar datos sobre personas represaliadas
  - Tiene una sección con noticias para dar a conocer la actividad del grupo.
- Módulo privado
  - Permite a los investigadores ver y modificar la base de datos de forma sencilla.
  - Capacidad para hacer búsquedas avanzadas con el fin de realizar estudios.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

LÍNEA DE TRABAJO

- Diseño de la base de datos en función de las necesidades de los investigadores
  - Realizar reuniones periódicas con Juan Carlos y Gotxon

- Elaboración de una web con las funcionalidades requeridas por el grupo de investigación y una estética propia
  - Familiarizarse con phpCan

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INTRODUCCIÓN A PHPCAN

- Framework basado en PHP5
- Estructura basada en la programación orientada a objetos.
- Proporciona un conjunto de funciones y clases que nos permiten evitar la programación a nivel de PHP o HTML.
- Utiliza “formats”

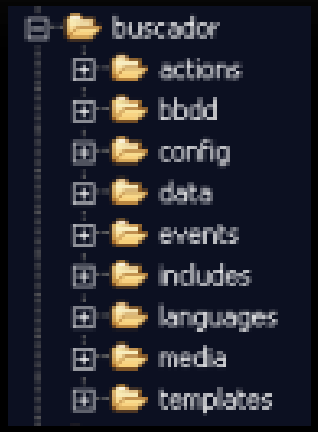
upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

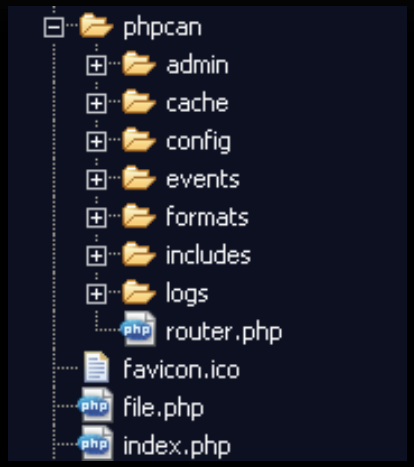
INTRODUCCIÓN A PHPCAN

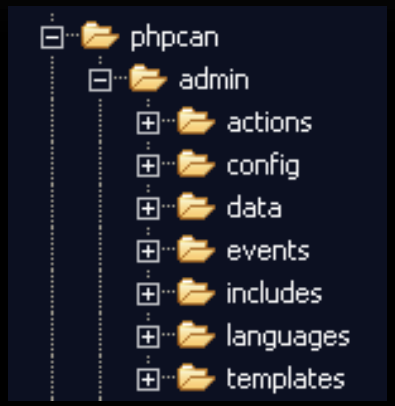
- Explicación de las clases más importantes.
  - Clase Config
    - Se encarga de almacenar la configuración de la web
    - Almacena en arrays datos de las demás clases.
  - Clase templates
    - Establece la estructura de los archivos específicos para el diseño de la interfaz.
  - Clase data
    - Proporciona métodos para la comunicación con la base de datos.
  - Clase html
    - Permite la creación de elemento HTML

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

ESTRUCTURA







upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresaibatu dira

CARPETA CONFIG

- Los archivos archivos data.php, sessions.php, actions.php y templates.php son comunes en ambas carpetas.
- El archivo tables.php se encarga de la configuración de la base de datos
- Bd.php se encarga configurar la comunicación con la base de datos
- El archivo base.php se encarga de algunas características básicas de la web.
- Paths.php se encarga de fijar las rutas a diversas carpetas de nuestra web.
- El archivo Languages.php establece los idiomas disponibles de nuestra web.

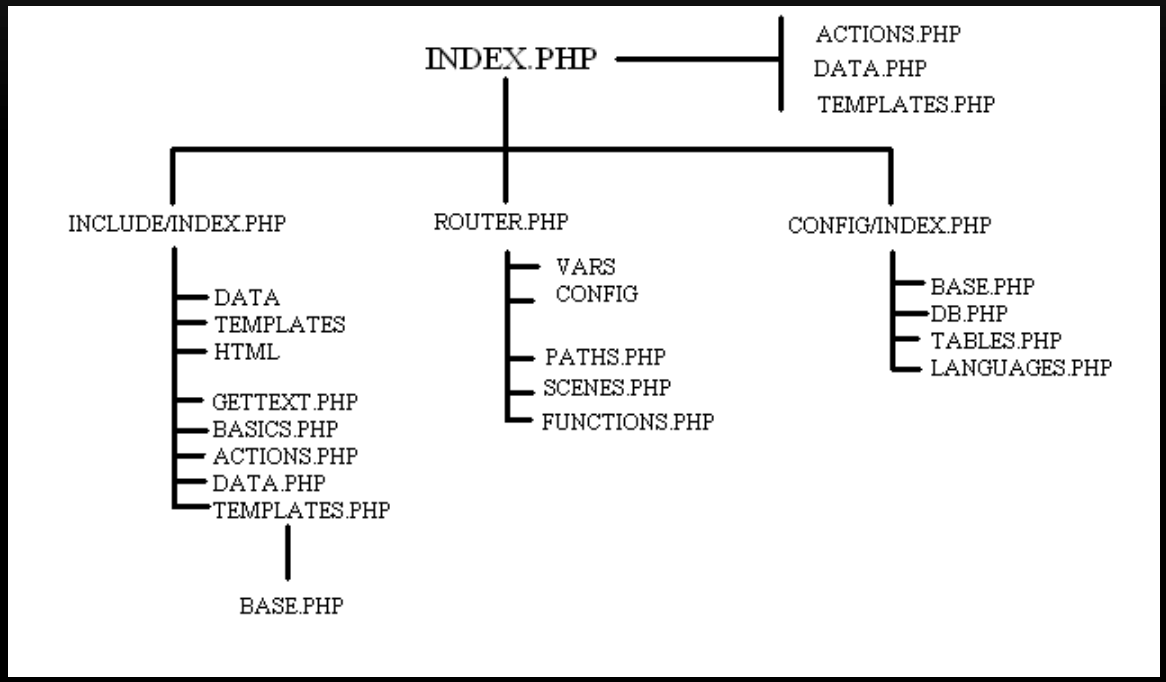
upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

DISEÑO DE LA BASE DE DATOS

- Trabajamos con entidades y relaciones
- Guardamos la definición de las entidades en un atributo de la clase *config* (*$tables*).
  - Utilizamos *formats para la definición de los campos.*
- Guardamos la configuración de las relaciones en el atributo *$relations de la misma clase.*
  - Los distintos tipos de relaciones afectan de forma diferente a nuestras tablas.
  - Usamos los parámetros tables, code, name y relation_table.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

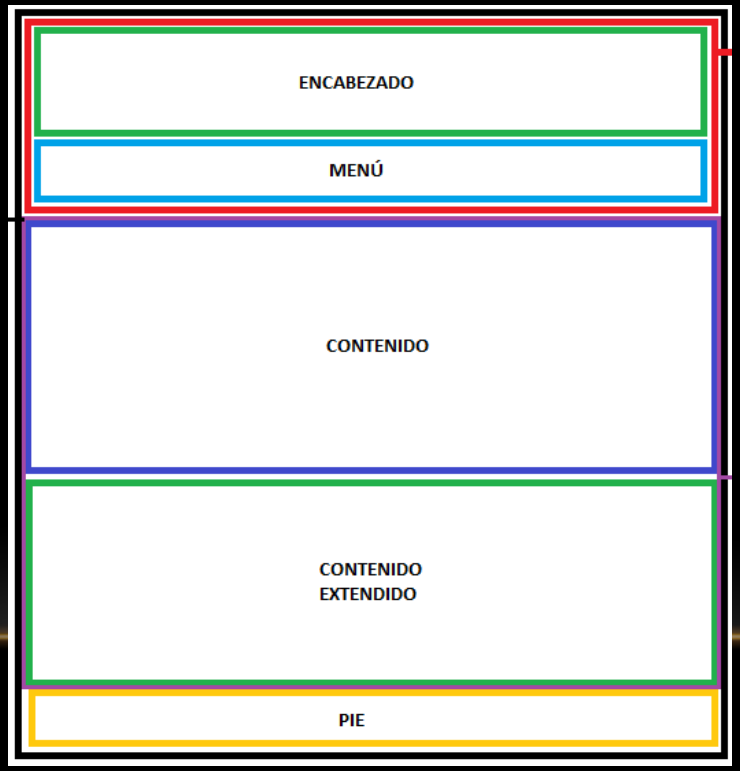
PRIMEROS PASOS



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CREACIÓN DE LA WEB

- Nos centramos en la parte pública de nuestra web.
- La web tiene que ser multiidioma y fácil de utilizar
- Utilizamos el archivo base.php como punto de partida.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CABECERA Y ENCABEZAMIENTO

- Cabecera
  - Cargamos los archivos .css y .js con el comando headlinks()

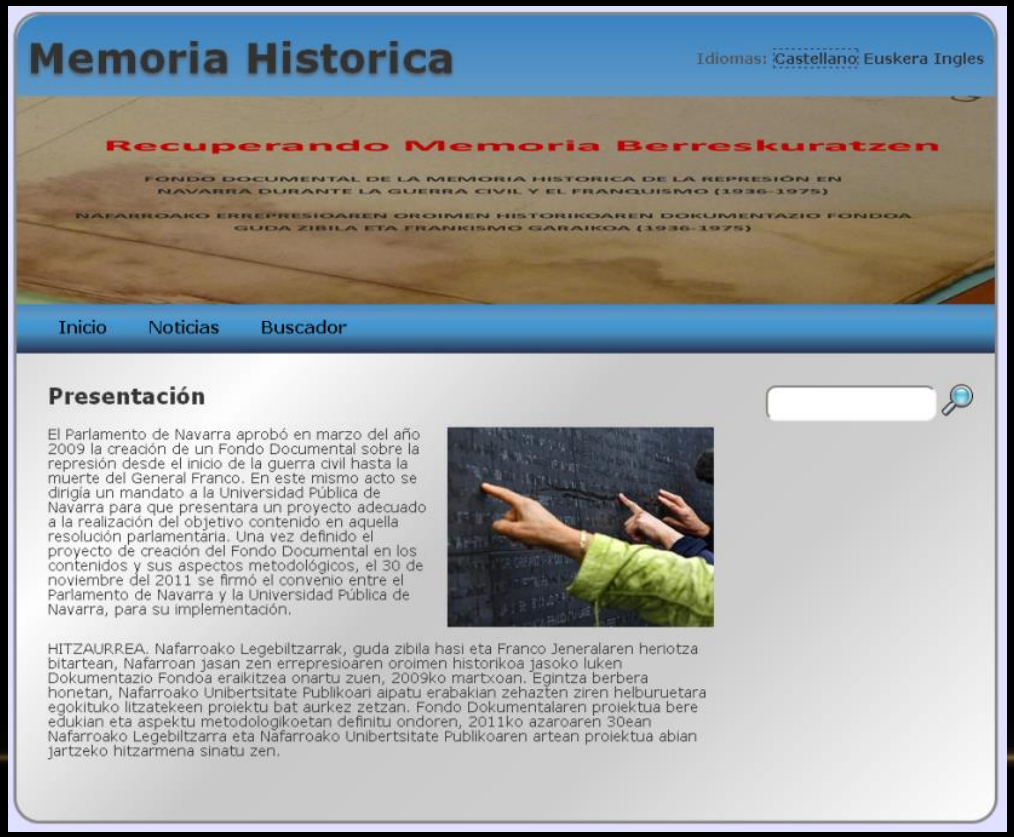
- Encabezamiento

  - Se compondrá de un título y una imagen principal
  - Dispone de un selector de idiomas.
  - Incluye un menú por el que desplazarse.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

INICIO

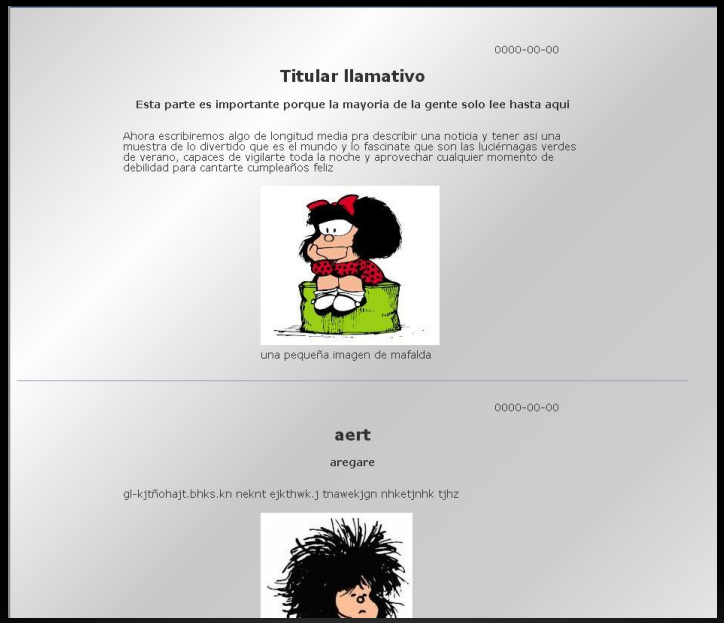
- Debe tener una presentación en euskera y en inglés.
- Tiene que disponer de un buscador simple.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

NOTICIAS

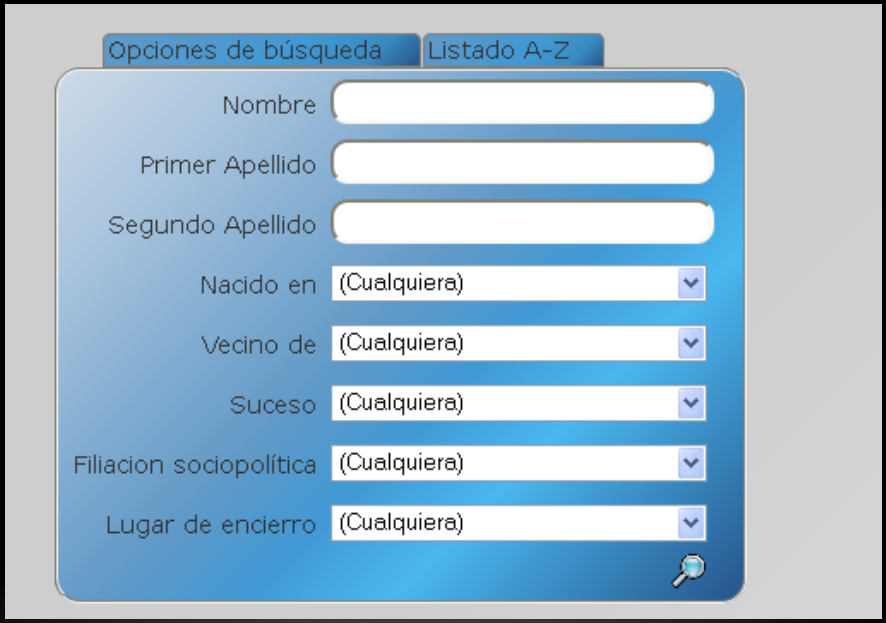
- Listado de noticias.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

BUSCAR

- Buscador
  - Una pestaña desplegable y otra para listar todo.
  - Un cajetín para la búsqueda simple.

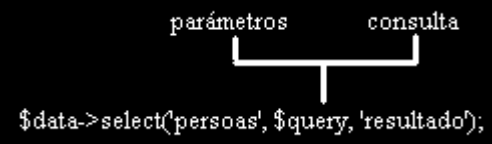


upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

BUSCAR

- Datos_buscador.php
  - Hace consultas a las bases de datos que correspondan y se las pasa a web_buscador.php.

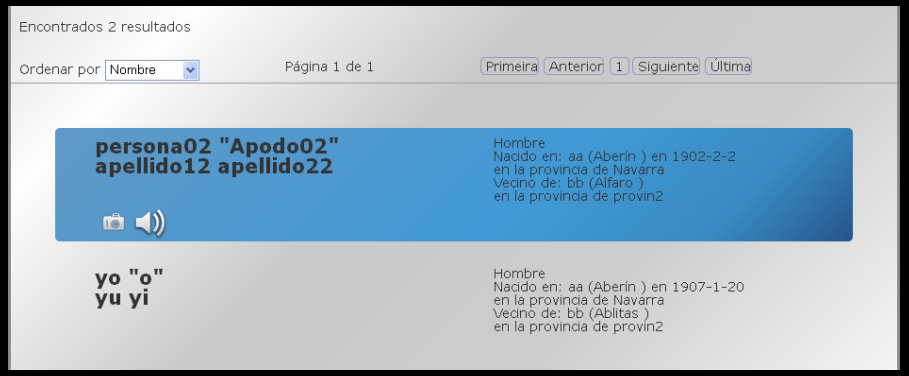
- Buscar.php
  - Guarda los datos procesados como parámetros para la búsqueda.
  - Junta en una sola variable la consulta con los parámetros mediante el método like() de la clase search.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

LISTADO

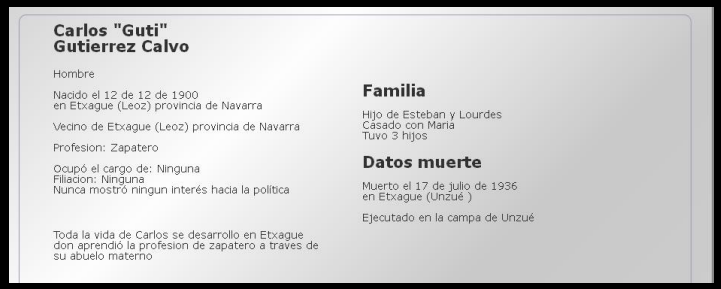
- Utiliza los registros de la consulta mostrar una pequeña ficha de todas las personas encontradas.
- Proporciona un navegador con el que desplazarte por los resultados
- Permite ordenar la lista en función del nombre, primer apellido o fecha de nacimiento.



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresaibatu dira

FICHA PERSONAL

- Ofrece información detallada sobre la persona solicitada



upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresaibatu dira

TRADUCCIONES

- Empleamos las funciones __() y __e().
- Creamos un archivo .po con todas las cadenas de texto que queramos traducir
- Utilizamos el programa poedit para transformar el archivo en uno .mo
- Subimos el archivo .mo a su correspondiente carpeta dentro de la carpeta languajes.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

VOLCADO DE ARCHIVOS CSV

- Nos aseguramos que el archivo excel incluya los campos necesarios en el orden correcto.
- Transformamos el archivo excel en un archivo csv separado por comas.
- Subir el archivo a la carpeta bbdd .
- Comprobamos que la codificación del archivo es utf-8.
- Cargar el archivo mediante el comando: LOAD DATA LOCAL INFILE

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

CONCLUSIONES

- El trabajar para otras personas para otras personas fue una experiencia novedosa.
- El aprendizaje de un nuevo lenguaje de programación siempre es un reto interesante.

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira

SISTEMA DE INFORMACIÓN DE REPRESALIADOS DE LA GUERRA CIVIL EN NAVARRA

Iosu Abaurrea Roldan

upna Universidad Pública de Navarra Nafarroako Unibertsitate Publikoa
Todos los derechos reservados
Eskubide guztiak erresalbatu dira